Apuntes U 3

Tema ***Divide y vencerás, Recursividad, aritmética Modular, MCD, Criptografía***

"Los sabios buscan la sabiduría; los necios creen haberla encontrado"
Napoleón

Divide y vencerás

Puede considerarse como una filosofía para resolver problemas. Es una técnica para el diseño de algoritmos que consiste en resolver un problema a partir de la solución de sub-problemas del mismo tipo, pero de menor tamaño.
Si los sub-problemas son todavía relativamente grandes se aplicará de nuevo esta técnica hasta alcanzar sub-problemas lo suficientemente pequeños para ser solucionados directamente.

Pasos para aplicar la técnica de "divide y vencerás":

1- Plantear el problema de forma que pueda ser descompuesto en k sub-problemas del mismo tipo, pero de menor tamaño. Es decir, si el tamaño de la entrada es n, hemos de conseguir dividir el problema en k sub-problemas (donde $1 \leq k \leq n$), cada uno con una entrada de tamaño nk y donde $0 \leq nk < n$. A esta tarea se le conoce como división.
   El número k debe ser pequeño e independiente de la entrada determinada, por ejemplo: en el caso particular de contener una sola llamada recursiva tendremos $k = 1$
   Donde llegamos a este tipo de algoritmo, podemos hablar de simplificación; ejemplos: el cálculo de la factorial de un número. También son algoritmos de simplificación el de búsqueda binaria en un vector o el que resuelve el problema del k-ésimo elemento.
2- En segundo lugar han de resolverse independientemente todos los sub-problemas, bien directamente si son elementales o bien de forma recursiva. El hecho de que el tamaño de los sub-problemas sea estrictamente menor que el tamaño original del problema nos garantiza la convergencia hacia los casos elementales, también denominados casos base.
3- Por último, combinar las soluciones obtenidas en el paso anterior para construir la solución del problema original.

¿Que NO es la recursividad?

Primero debemos decir que la recursividad no es una estructura de datos, sino que es una técnica de programación que nos permite que un bloque de instrucciones se ejecute **n** veces. Remplaza en ocasiones a estructuras **repetitivas o iterativas**.

¿Qué es?

Técnica de programación muy potente que puede ser usada en lugar de la iteración.

Algoritmo recursivo

Un algoritmo recursivo es un algoritmo que expresa la solución de un problema en términos de una **llamada a sí mismo**. La llamada a sí mismo se conoce como llamada recursiva o recurrente.

```
FUNCIÓN Factorial(n)
    VAR resultado: Entero

    SI (n<2) ENTONCES
        resultado = 1;
    SINO
        resultado = n * Factorial(n-1);
    FSI

    RETORNA resultado;
FUNCIÓN
```

La recursividad es uno de los conceptos más importantes en programación y es una capacidad que tienen los sub-programas a auto invocarse.
La recursividad nos permite trabajar donde las estructuras del tipo iterativas vistas en la Unidad 1 no son de fácil aplicación. Esto nos indica que **no hay problemas intrínsecamente recursivos o iterativos**; cualquier proceso puede expresarse de una u otra forma.

Si bien la recursividad, como veremos, da lugar a algoritmos más fáciles de seguir y entender, debemos tener en cuenta que consume más recursos y es más lenta que una solución iterativa.
Para desarrollar algoritmos recursivos hay que partir del supuesto de que ya existe un algoritmo que resuelve una versión más sencilla del problema.
A partir de esta suposición debe hacerse lo siguiente:

1. Identificar sub-problemas atómicos de resolución inmediata (casos base).
2. Descomponer el problema en sub-problemas resolubles mediante el algoritmo pre-existente; la solución de estos sub-problemas debe aproximarnos a los casos base.
3. Probar de manera informal que tanto los casos base como los generales pueden solucionarse con el algoritmo desarrollado.

Se pueden presentar uno o varios, nuevamente dependiendo del problema a resolver.
Cuando se analiza la solución recursiva de un problema es importante determinar con precisión cuáles serán los pasos básico y recursivo.
En cada vuelta del ciclo es importante que nos acerquemos cada vez más a la solución del problema en cuestión, es decir, al paso básico. Si esto no ocurre podemos estar en un ciclo extraño.
Puede que el problema se encuentre mal definido y entonces la máquina se quedaría ejecutando por tiempo indefinido el programa y sólo terminaría al agotarse la memoria.
Como todas las funciones recursivas tienen la misma estructura, el cuerpo de la función será un condicional.

¿Cuántas condiciones debo poner?
- Una por cada caso base
- Una por el caso recursivo
Se debe probar primero el caso base, porque si éste tiene errores (lógicos) es posible que la función se quede haciendo un ciclo infinito.

Tipos de recursividad

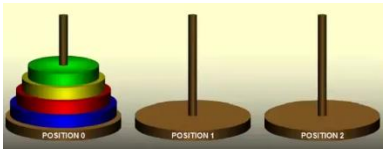
- **Directa:** el programa o subprograma se llama directamente a si mismo.
- **Indirecta:** el programa o subprograma llama a otro y éste en algún momento llama al primero.

Claves de la recursividad

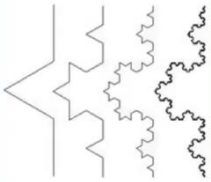
Los compiladores y/o intérpretes deben tener características de aceptar recursividad (en la actualidad todos o la gran mayoría lo tienen).
La recursividad nos permite trabajar donde las estructuras del tipo iterativas vistas en la no son de fácil aplicación. Esto nos indica que no hay problemas intrínsecamente recursivos o iterativos; cualquier proceso puede expresarse de una u otra forma.

Problemas típicos de recursividad

- **Torres de Hanói**



- **Fractales**



- **Sumatorias**

$$S = \sum_{i=0}^{N} a_i$$

- **Sucesiones de Fibonacci**

- **Factorial**

Proceso de recursividad con el cálculo factorial

Supongamos que calcularemos el factorial de 4 (4!).
Sabemos matemáticamente que

$$4! = 4 \times 3 \times 2 \times 1 = 24$$

También que

$$3! = 3 \times 2 \times 1$$

y que

$$2! = 2 \times 1$$

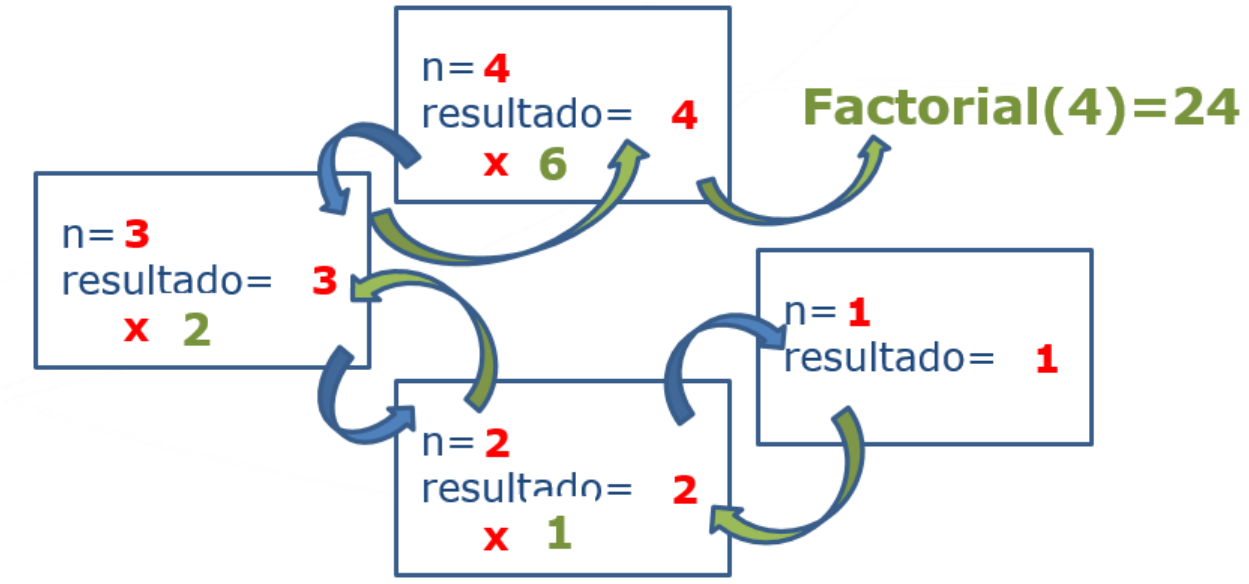
Y por definición

$$1! = 1$$

¿Podemos encontrar un patrón?

Entonces también podemos expresarlo de la siguiente forma:

$$4! = 4 \times 3!$$
$$3! = 3 \times 2!$$
$$2! = 2 \times 1!$$
$$1! = 1$$



Ejemplos de código de ña función factorial recursiva

```
FUNCIÓN Factorial(n)
    VAR resultado: Entero

    SI (n<2) ENTONCES
        resultado = 1;
    SINO
        resultado = n * Factorial(n-1);
    FSI

    RETORNA resultado;
FUNCIÓN
```

```
1  package recursividad;
2
3  public class factorialrecursivo {
4      public int CalcularFactorial (int numero){
5          if  (numero <2){
6              return 1;
7          } else {
8              return numero * CalcularFactorial (numero-1);
9          }
10
11     }
12
13 }
```

Sucesión de Fibonacci (recursiva)

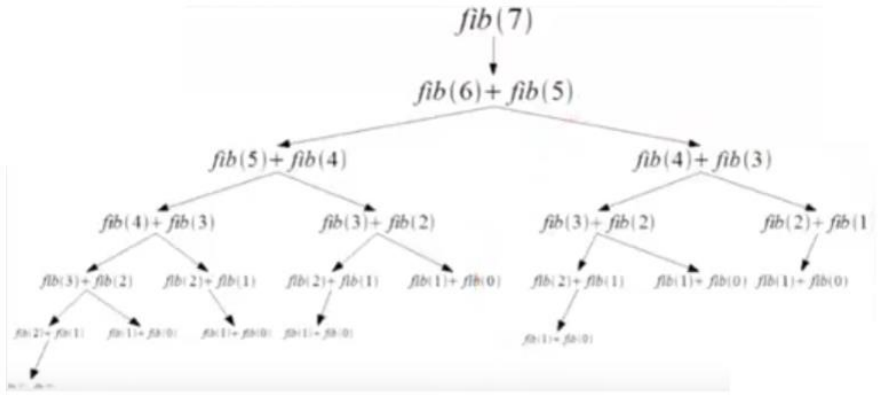
$$0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597 \ldots$$

La sucesión comienza con los números 0 y 1, y a partir de estos, «cada término es la suma de los dos anteriores», es la relación de recurrencia que la define.
A los elementos de esta sucesión se les llama números de Fibonacci. Esta sucesión fue descrita en Europa por Leonardo de Pisa, matemático italiano del siglo XIII también conocido como Fibonacci. Tiene numerosas aplicaciones en ciencias de la computación, matemática y teoría de juegos. También aparece en configuraciones biológicas, como por ejemplo en las ramas de los árboles, en la disposición de las hojas en el tallo, en las flores de alcachofas y girasoles, en las inflorescencias del brécol romanesco y en la configuración de las piñas de las coníferas. De igual manera, se encuentra en la estructura espiral del caparazón de algunos moluscos, como el nautilus.

Ter	Sucesión(n)
0	0
1	1
2	1
3	2
4	3
5	5
6	8
7	13
8	21
9	34
10	55

Casos Base

$$f_n = f_{n-1} + f_{n-2}$$



Técnica para implementar recursividad

- Para desarrollar algoritmos recursivos hay que partir del supuesto de que ya existe un algoritmo que resuelve una versión más sencilla del problema..
- A partir de esta suposición debe hacerse lo siguiente:

1. Identificar sub-problemas atómicos de resolución inmediata (***casos base***).
2. Descomponer el problema en sub-problemas resolubles mediante el algoritmo pre-existente; la solución de estos sub-problemas debe aproximarnos a los casos base.
3. Probar de manera informal que tanto los casos base como los generales pueden solucionarse con el algoritmo desarrollado.

Nunca debemos resolver la misma instancia del problema en llamadas recursivas separadas

Demostración por inducción matemática

Se trata de una técnica de demostración que se utiliza para demostrar muchos teoremas que afirman que P(n) es verdadera para todos los enteros positivos n.
En matemáticas, la inducción es un razonamiento que permite demostrar una infinidad de proposiciones, o una proposición que depende de un parámetro n que toma una infinidad de valores enteros. En términos simples, la inducción matemática consiste en el siguiente razonamiento:

* Premisa mayor: El número entero tiene la propiedad P.*
* Premisa menor: El hecho de que cualquier número entero n tenga la propiedad implica que también n+1 la tiene.

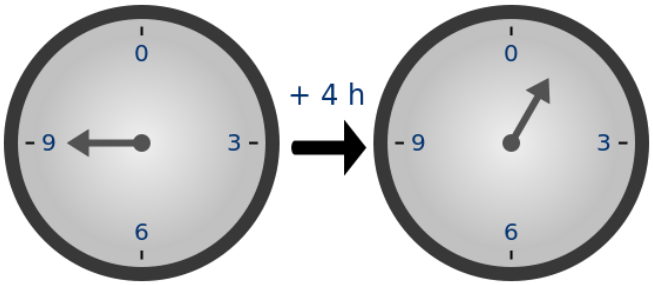
Conclusión: Todos los números enteros a partir de a tienen la propiedad P. Con más rigor, el método de inducción matemática es el que realiza la demostración para proposiciones en las que aparece como variable un número natural. Se basa en un axioma denominado principio de la inducción matemática.

Una descripción informal de la inducción matemática puede ser ilustrada por el efecto dominó, donde ocurre una reacción en cadena con una secuencia de piezas de dominó cayendo una detrás de la otra.

PASO BASE: Se muestra que la proposición P(1) es verdadera.
PASO INDUCTIVO: se muestra que la implicación P(k) -> P(k + 1) es verdadera para todo entero positivo k

Aritmética Modular

En matemática, la aritmética modular es un sistema aritmético para clases de equivalencia de números enteros llamadas clases de congruencia. La aritmética modular fue introducida en 1801 por Carl Friedrich Gauss.
Algunas veces se le llama, sugerentemente, aritmética del reloj, ya que los números «dan la vuelta» tras alcanzar cierto valor llamado módulo.



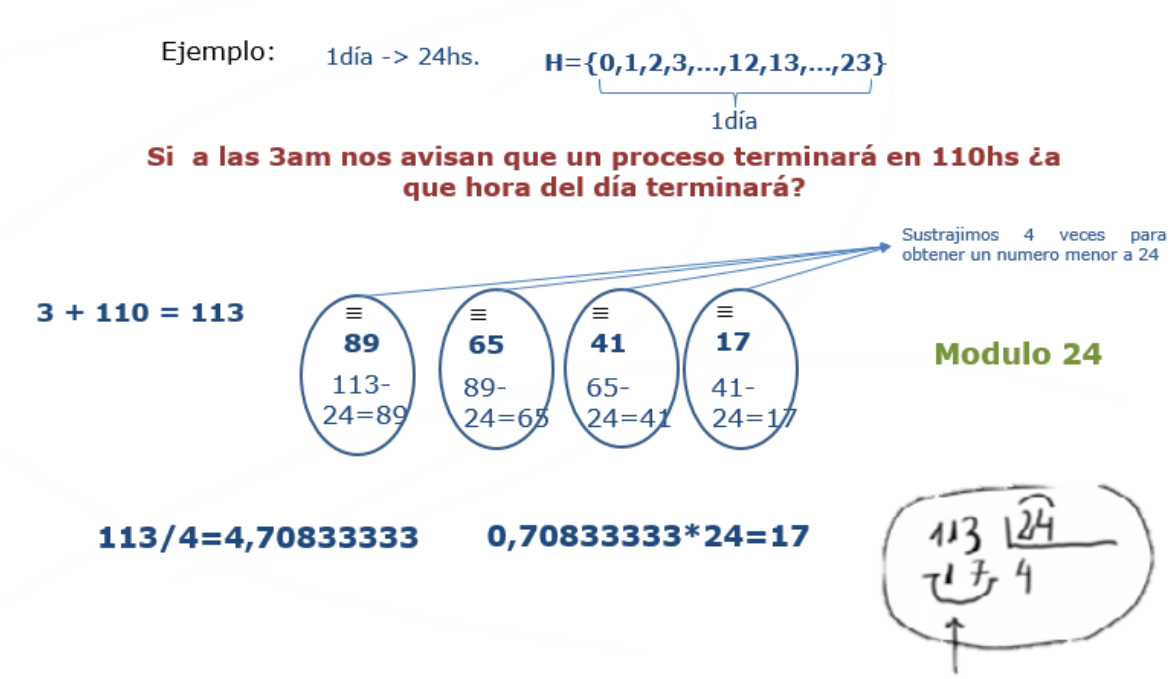
La Aritmética Modular es una de las aplicaciones más importantes de la teoría de números. Y está representada por la función mod.
La función módulo representa el resto de la división. Por ejemplo a mod b significa que queremos hallar el resto de a, que representamos como: a mod b = a – b {a/b}

La Aritmética Modular también define un sistema de numeración.
Veamos por ejemplo la secuencia: 0 mod 3, 1 mod 3, 2 mod 3, 3 mod 3, 4 mod 3....
Evaluando tenemos 0, 2, 1, 0, 1... que equivale a la numeración en base 3.

Algunas de las propiedades más importantes de la Aritmética Modular.
Suma, resta, multiplicacion, Division (no esta definida para todos los caos)

Existen muchas aplicaciones de la Aritmética Modular, por ejemplo en el calendario los días de la semana corresponden a una aritmética módulo 7, las horas, minutos y segundos corresponden al módulo 60. Hallar el último dígito de un número decimal corresponde a una aritmética módulo 10.

Ejemplo de cómo ayudaría



Cuestiones Prácticas

Está representada por la función mod (% en Java).
La función módulo representa el resto de la división y queda definida por $a \bmod b = a - b\{a/b\}$.

Propiedades:

Suma: **(x + y) mod m = (x mod m + y mod m) mod m**.

Resta: La resta es sólo la suma con valores negativos por los que **(x – y)**

mod m = (x mod m – y mod m) mod m.

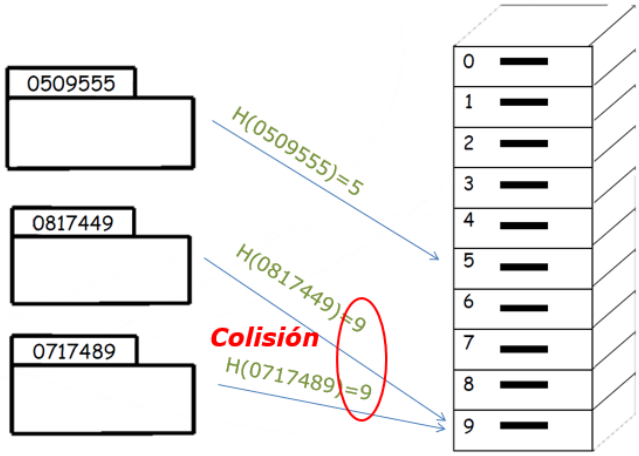
Multiplicación: La multiplicación **xy mod m = (x mod m)(y mod m) mod m**.
Esto se da debido a que la multiplicación es simplemente una suma repetida.

División: No existe el inverso de la multiplicación genérico. *(pero existen algoritmos que permiten su cálculo en los casos que si existe)*

Congruencia o Equivalencia: $a \equiv b(mod\ n)$ Se lee como que **a** es equivalente a **b modulo n**.

$$a_1 \equiv b_1(mod\ n)$$
$$a_2 \equiv b_2(mod\ n)$$
$$a_1 + a_2 \equiv b_1 + b_2(mod\ n)$$
$$a_1 * a_2 \equiv b_1 * b_2(mod\ n)$$

Algunas de sus aplicaciones son en Tabla Hash y en criptografía.
Por ejemplo Dado un código k, para conocer el sitio donde se almacena, se utiliza la función: **h(k)= k mod 10** .



Exponenciación modular

El operador es el equivalente aritmético a calcular una exponenciación natural y luego calcular el modulo o residuo del resultado respecto a cierto número (M).
La Exponenciación Modular recibe tres operandos de entrada:

• El operando X es llamado comúnmente la base de la exponenciación,
• El numero Y es referido como el exponente
• M es referido como el modulo de la representación.

Definición matemática:

Sean X, Y y M números enteros tal que: $M > 0$.
Se define la Exponenciación Modular P como:

$P \equiv X^Y \ mod \ M$ si y solo si existe un número entero k tal que:
$X^Y = k.M + P$

Propiedad recursiva:

$$\mathbf{x = a^b mod\ n = (a\ mod\ n)a^{b-1} mod\ n}$$

Ejemplo:

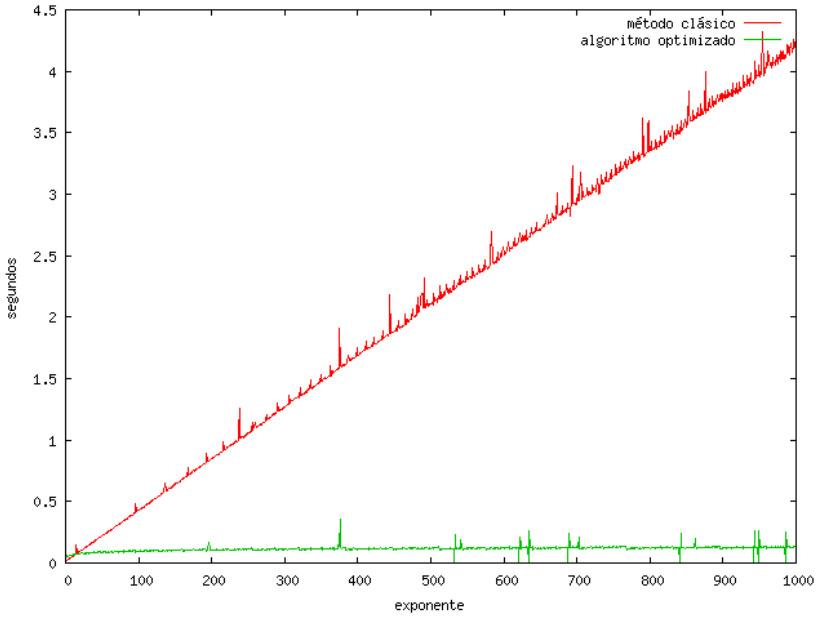
Calculo normal: $4^5 mod\ 7 = 1024\ mod\ 7 = 2$

Calculo recursivo: $4\ mod\ 7 = 4$; $(4 * 4)\ mod\ 7 = 2$; $(2 * 4)\ mod\ 7 = 1$; $(1 * 4)\ mod\ 7 = 4$; $(4 * 4)\ mod\ 7 = 2$ *(implementado 5 veces)*

La Exponenciación Modular es un operador de amplio uso en técnicas criptográficas tales como:
• RSA • Rabin • Elgamal • McEliece
Básicamente, el operador es el equivalente aritmético a calcular una exponenciación natural y luego calcular el modulo o residuo del resultado respecto a cierto número (M).

La Exponenciación Modular debe operar sobre números enteros de gran tamaño, generalmente en binario, esto es debido a que por niveles de seguridad cada vez más exigentes en los sistemas criptográficos modernos, obliga a un aumento paulatino en el tamaño en bits de las claves, de modo que los operadores que componen el sistema deben manejar operandos de mayor tamaño.
Esta situación tiende a hacer lenta la ejecución de algunos de estos operadores, especialmente en aquellos de gran complejidad.
Existe entonces un interés por la optimización de tales operaciones, de modo que se puedan conseguir tiempos de ejecución aceptables.
Las implementaciones en hardware aparecen como alternativa de diseño, ya que el hardware dedicado y optimizado para una operación es potencialmente más rápido que la ejecución de una aplicación de software sobre hardware genérico.

La idea detrás de la exponenciación modular rápida consiste en obtener la representación del exponente n en dígitos binarios (dt,dt-1,...,d2,d1), con dt =1, y hallar los distintos cuadrados sucesivos (mod m) de la base a: (a1,a2,a4,...,a2t), para después multiplicar módulo m las potencias a2i correspondientes a los dígitos binarios di que sean "1".



Complejidad de algoritmos recursivos

Un aspecto interesante de este ejemplo en particular es que la **complejidad** del algoritmo recursivo para el cálculo del factorial es idéntica a la del algoritmo iterativo pues en ambos casos es O(n).

¿Quiere esto decir que ambos algoritmos son igual de eficientes?
No, recordemos que esto es un límite asintótico y prescindiendo de constantes multiplicativas.

Así, ambos algoritmos se comportarán de forma similar: si m es 2n, entonces el cálculo de m! tardará como máximo el doble que n!
Sin embargo, es posible que el algoritmo recursivo tarde más que el iterativo por cuestiones de la implementación de la recursividad en una computadora, no por cuestiones algorítmicas.

Pequeño Teorema de Fermat

Pierre Fermat fue un abogado que vivió en Francia de 1601 a 1665. Se interesó por las matemáticas cuando ya estaba en los treintas y siempre se acercó a ella como pasatiempo; sin embargo, los pasatiempos de los genios son superiores a toda una vida de dedicación de gentes menos dotadas. Trabajó mucho con Teoría de los Números, una rama de las matemáticas que podemos identificar con la Aritmética, en particular con los relojes de Gauss, y realizó algunos descubrimientos importantes.

> Si p es un número primo, entonces, para cada número natural a, con $a>0$, $a^p \equiv a \pmod p$
>
> Pierre de Fermat, 1636

Aunque son equivalentes, el teorema suele ser presentado de esta otra forma:

> Si p es un número primo, entonces, para cada número natural a, con $a>0$, coprimo con p , $a^{p-1} \equiv 1 \pmod p$
>
> Pierre de Fermat, 1636

Esto quiere decir que, si se eleva un número **a** a la **p-ésima** potencia y al resultado se le resta **a**, lo que queda es divisible por **p** *(aritmética modular)*. Su interés principal está en su aplicación al problema de la primalidad y en criptografía.

Este teorema no tiene nada que ver con el legendario último teorema de Fermat, que fue sólo una conjetura durante 350 años y finalmente fue demostrado por Andrew Wiles en 1995.

MCD (Máximo Común divisor)

En matemáticas, se define el máximo común divisor *(abreviado mcd o "M.C.D")* de dos o más números enteros al mayor número que los divide sin dejar resto.

Hay dos formas de hallar el mcd, tenemos una forma larga , que sería buscando todos los divisores de los números y lo mismo en el caso de los múltiplos, hasta encontrar el común. Pero de manera mucho mas rápida se puede resolver mediante la división por números primos.

Esta es la forma larga *(o para cuando aún no se conocen los números primos)*, de buscar el **mcd**. Buscamos los divisores de cada número, y seleccionamos entre todos los divisores comunes *(que se repiten en los dos grupos)* el mayor de ellos será el **mcd**.

Div (12)____{1, 2, 3, 4, 6, 12}

Div (18)____{1, 2, 3, 6, 9, 18}

m.c.d (12,18)= 6

Una de las formas cortas de llegar al mismo resultado es hallando el **mcd** de ambos números para ello tomamos solamente los que son comunes en los dos, en este ejemplo hemos seleccionado solo un 2 porque solo uno es común y un 3 por la misma razón, el 5 no lo utilizamos porque no es común en los dos, y realizamos la multiplicación (2.3=6).

12	2
6	2
3	3
1	

12= 2.2 . 3

18	2
9	3
3	3
1	

18= 2. 3. 3

m.c.d (12,18)= 2 . 3= 6

Descomponemos en factores primos los dos números y tomamos los factores comunes a ambos con el menor exponente con el que aparezcan.
El problema inicial es el siguiente:
Encontrar el máximo común divisor entre dos números enteros positivos a y b.

Aunque es un método bastante útil y sencillo para conseguir lo que queremos tiene un evidente problema: si los números son muy grandes, o si sus factores primos lo son, se complica ya que el cálculo de la descomposición se torna bastante tedioso.

MCD – Algoritmo de Euclides

El **algoritmo de Euclides** nos servirá para esta tarea:

- Tomando **a** y **b**, dividimos el más grande por el mas pequeño *(suponemos a el mas grande y b el más pequeño)*; y nos proporcionará un cociente, **c1**, y un resto, **r1**.
- Si r1=0 , entonces mcd(a,b)=b. *(asunto resuelto)*
- Si no es cero dividimos el divisor, c1, entre el resto, r1, obteniendo otro cociente,c2, y otro resto,r2.
- Si r2 = 0, entonces mcd(a.b) = r1 .
- Si no es cero volvemos a dividir divisor entre resto.

Y así sucesivamente.

Esto es, el máximo común divisor entre a y b es el último resto distinto de cero que obtengamos con el procedimiento anterior.
Si analizamos el algoritmo de Euclides se ve claramente que necesitamos demostrar que el máximo común divisor entre a y b es igual al máximo común divisor entre b y r 1 .
De este modo, esa igualdad se mantendrá durante todo el proceso y llegaremos a que el último resto distinto de cero es el máximo común divisor de los dos enteros positivos iniciales.

Teorema:

- El máximo común divisor de dos números enteros positivos a y b, con a > b > 0, coincide con el máximo común divisor de b y r, siendo r el resto que se obtiene al dividir a entre b.

Demostración:

Sean $d = mcd(a, b)$ y $t = mcd(b, r)$

Vamos a demostrar que $d = t$.

Por definición de máximo común divisor, se tiene que d es un divisor tanto de a como de b. Por tanto $a = a_1 d$ y $b = b_1 d$

Por otro lado, por el algoritmo de la división se tiene que:

$a = bq + r$, con $r < b$ (1)

Por tanto d es un divisor de r. Como ya teníamos que también es un divisor de b entonces debe dividir a su máximo común divisor, esto es, d **es un divisor de** t.

Por otro lado, t es un divisor tanto de b como de r. Por ello se tiene que $b = pt$ y r=st. Sustituyendo estas dos igualdades en (1) obtenemos lo siguiente:

$$a = ptq + st = (pq + s)t$$

Por tanto t es un divisor de a. Como también lo era de b debe ser un divisor de su máximo común divisor, es decir, t es un divisor de d.

Como d es un divisor de t y t es un divisor de d no queda otra opción más que $\boldsymbol{t = d}$.

Como ya teníamos que también es un divisor de b entonces debe dividir a su máximo común divisor, esto es, d es un divisor de t.
Por otro lado, t es un divisor tanto de b como de r.
Por ello se tiene que b = pt y r = st .
Sustituyendo estas dos igualdades en (1) obtenemos lo siguiente:

a = ptq + st = (pq + s)t

Por tanto t es un divisor de a. Como también lo era de b debe ser un divisor de su máximo común divisor, es decir, t es un divisor de d.
Como d es un divisor de t y t es un divisor de d no queda otra opción más que t = d.
Por tanto el algoritmo de Euclides funciona.

MCD – Ejemplo

Cálculo de mcd(721, 448)

Como hemos explicado antes dividimos el número mayor entre el menor; si el resto no es cero dividimos el divisor entre el resto; y así sucesivamente hasta que llegamos a un punto en el que el resto es cero. Los resultados de las divisiones (expresados como dividendo=divisor • cociente + resto) son:

		div	mod
721	448	1	273
448	273	1	175
273	175	1	98
175	98	1	77
98	77	1	21
77	21	3	14
21	14	1	7 *
14	7	2	0

Como marca el *, se tiene que mcd(721, 448) = 7, el último divisor que no es nulo.

Un ejemplo de un algoritmo que realice esta función puede ser:

```
funcion mcd-euclides (a,b:entero):entero;
        div,resto,restoant:entero;
        resto=1;
        {
        mientras (resto != 0) hacer
                        restoant=resto;
                        resto = a mod b;
                        a=b;
                        b=resto;
                fin mientras ;
        retornar restoant;
        }
```

	a	b	div	resto	restoant
1	721	448	1	273	1
2	448	273	1	175	273
3	273	175	1	98	175
4	175	98	1	77	98
5	98	77	1	21	77
6	77	21	3	14	21
7	21	14	1	7	14
8	14	7	2	0	7

Inverso Multiplicativo

El inverso multiplicativo, recíproco e inversa de un número ***x*** no nulo, es el número, denotado como $1/x$ o x^{-1}, que multiplicado por ***x*** da ***1*** como resultado.
El multiplicador modular inverso de un entero ***n*** módulo ***p*** es un entero ***m***, tal que:

$$n^{-1} \equiv m(mod\ p)$$

Esto significa que es el inverso multiplicativo en el anillo de los enteros módulo ***p***. Es equivalente a

$$m.n \equiv 1(mod\ p)$$

El multiplicador inverso de ***n*** módulo ***p*** existe si y sólo si ***n*** y ***p*** son **coprimos**, es decir, si MCD(n, p)=1.

> En matemáticas, dos números enteros a y b son números primos entre sí (o **coprimos**, o primos relativos), si no tienen ningún factor primo en común, o, dicho de otra manera, si no tienen otro divisor común más que 1 y -1. Equivalentemente son primos entre sí, si y sólo si, su máximo común divisor es igual a 1.

Si existe el multiplicador modular inverso de ***n*** módulo ***p***, se puede definir la operación de división entre ***n*** módulo ***p*** mediante la multiplicación por el inverso.

El 0 no tiene inverso multiplicativo. Todo número complejo, salvo el 0, tiene un inverso que es un número complejo. El inverso de un número real también es real, y el de un número racional también es racional.
El inverso multiplicativo es aplicable a distintos tipos de objetos matemáticos.

- La inversa de una matriz es otra matriz, denominada matriz inversa, que al multiplicarse por la original es igual a la matriz identidad.
- La inversa de una función es la resultante de despejar la variable independiente, convirtiéndola en dependiente. Gráficamente es un trazado paralelo a la recta diagonal y = x.
- En las Matemáticas Constructivas, para que un número real x tenga inverso, no es suficiente que sea falso que x = 0. Además, debe existir un número racional r tal que 0 < r < |x|.
- En Aritmética Modular, el inverso multiplicativo de x también está definido: es el número a tal que (a × x) mod n = 1. ***Sin embargo, este inverso multiplicativo sólo existe si a y n son primos entre sí***.

Por ejemplo, el inverso de 3 módulo 11 es 4, porque es la solución de (3 × x) mod 11 = 1. Un algoritmo empleado para el cálculo de inversos modulares es el Algoritmo extendido de Euclides.

Números Primos

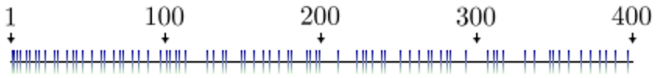
La razón más importante es la que está expresada en el Teorema fundamental de la Aritmética:

> **Todo número natural N puede expresarse como producto de números primos, N = p1p2…pn, y esta representación es única salvo por el orden.**

Por ejemplo, *4 = 2×2, 12 = 2x2x3, 500 = 100×5 = 2x50x5 = 2x5x10x5 = 2x5x2x5x2 = 2x2x2x5x5. Las dos últimas expresiones son la misma, salvo el orden en que están presentados los factores primos 2 y 5*.
Otro número para factorizar podría ser **1001 = 7×143 = 7x11x13**. Con este número batallamos un poquito. ¿Y con 6901? ¿Y con 280123?
Tomando en cuenta las dificultades para poder factorizar un número, podemos pensar que el teorema será todo lo fundamental que quieran los matemáticos, pero es de poca utilidad.

Ahora pensemos, si es tan difícil factorizar números, ¿entonces de qué nos sirve? Si alguien estuvo pensando previamente en esta objeción, acaba de descubrir el secreto de la criptografía.

Los números primos están metidos en la lista de los números naturales de una manera azarosa. Las claves no son no obvias, para que los vayamos descubriendo. Si hacemos una lista de los que nuestra paciencia nos permite, escribiremos algo así como 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, etc.



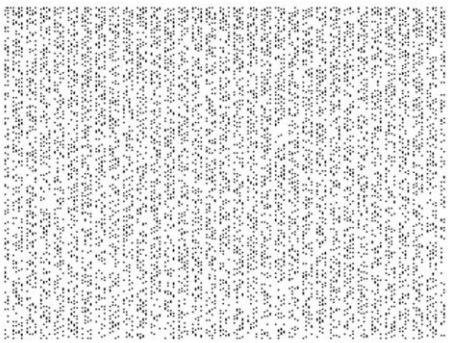
La distribución de los números primos (línea azul) hasta el 400.

¿Cuántos hay? ¿Es una lista infinita la de los números primos?

Un teorema tan viejo como la cultura griega nos dice que los **números primos son infinitos**. La prueba es por reducción al absurdo, es decir, se supone que la lista es finita, y de ahí se argumenta para llegar a una contradicción. El argumento es sencillo: supongamos que tenemos un número finito de primos p1, p2, p3,…pN. Consideremos ahora el número M = (el producto de todos los primos) + 1 = (p1 p2 p3…pN) + 1, y cuestionemos si M es primo o si no lo es. Es claro que M es mayor que cualquiera de los números primos, puesto que el producto de dos números mayores que 1 es mayor que cualquiera de los factores. Si M fuera primo, entonces tendríamos un número primo mayor que todos los números primos, en particular mayor que sí mismo, y eso es una contradicción.

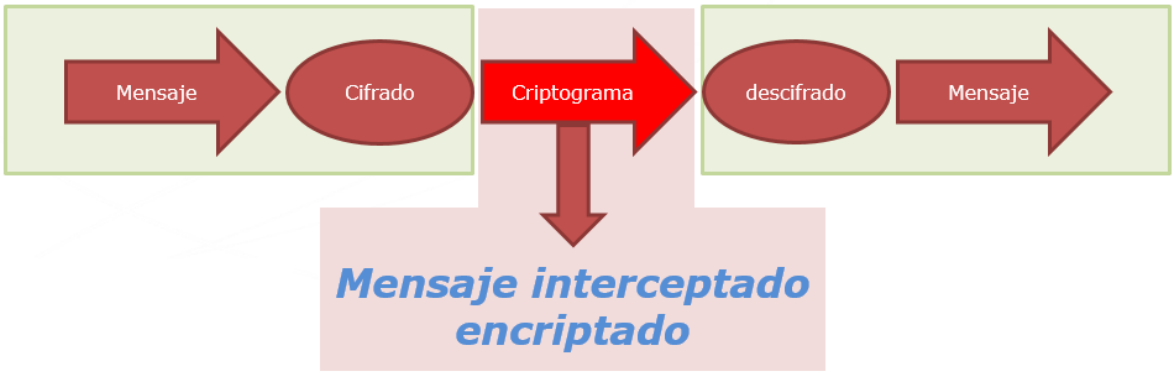
Si M no es primo, entonces tenemos una cantidad infinita de números especiales, los primos, distribuidos de una manera aparentemente aleatoria en la lista de todos los números naturales.

La **dificultad para factorizar un número cualquiera en sus factores primos** es precisamente la **ventaja** que utilizan los desarrolladores de algoritmos para codificar los números de tarjetas de crédito, para almacenar bases de datos encriptadas, etc.


La distribución de todos los números primos comprendidos entre 1 y 76.800, de izquierda a derecha y de arriba abajo. Cada pixel representa un número. Los píxeles negros representan números primos; los blancos representan números no primos.

Criptografía

es la ciencia de cifrar y descifrar información utilizando técnicas matemáticas que hagan posible el intercambio de mensajes de manera que sólo puedan ser leídos por las personas a quienes van dirigidos.



Criptografía en la Historia

Escitala Espartana

•Usada en la antigua Grecia en el año 400a.c
•Se enrolla una cinta sobre un vara
•El ancho con el cual fue escrito el mensaje corresponde con la vara adecuada para descifrar el mensaje

Una escítala (griego: skytálē) es un sistema de criptografía utilizado por los éforos espartanos para el envío de mensajes secretos. Está formada por dos varas de grosor variable (pero ambas de grosor similar) y una tira de cuero o papiro, a las que ambas se puede denominar escítala.
El sistema consistía en dos varas del mismo grosor que se entregaban a los participantes de la comunicación. Para enviar un mensaje se enrollaba una cinta de forma espiral a uno de los bastones y se escribía el mensaje longitudinalmente, de forma que en cada vuelta de cinta apareciese una letra de cada vez. Una vez escrito el mensaje, se desenrollaba la cinta y se enviaba al receptor, que sólo tenía que enrollarla a la vara gemela para leer el mensaje original.

Método Julio Cesar

Método Julio Cesar (para codificar)

1. *Transforme cada letra a un número, para ello, utilice la posición relativa en el alfabeto. A es 0, B es 1, C es 2 …*
2. *Aplique la función f(p)=(p+3) mod26 para cada número*
3. *Transforme cada número a letra y envíe el mensaje*

Para decodificar

1. *Transforme cada letra a número.*
2. *Utilice la función f-1(p)=(p-3) mod26*
3.

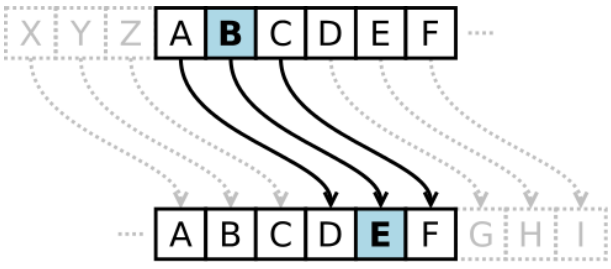
A	0	N	13
B	1	O	14
C	2	P	15
D	3	Q	16
E	4	R	17
F	5	S	18
G	6	T	19
H	7	U	20
I	8	V	21
J	9	W	22
K	10	X	23
L	11	Y	24
M	12	Z	25

Ejemplo:

- *Encriptar el mensaje "HOLA"*

f(p)=(p+3) mod26
f(7) = 10 mod26= 10
f(14) = 17 mod26= 17
f(11) = 14 mod26= 14
f(0) = 3 mod26= 3

- *El mensaje encriptado es "KROD"*



A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Máquina Atbash

Atbash es un método muy común de cifrado del alfabeto hebreo. Pertenece a la llamada criptografía clásica y es un tipo de cifrado por sustitución. Se le denomina también método de espejo, pues consiste en sustituir la primera letra (álef) por la última (tav), la segunda ([p|בbet]]) por la penúltima (shin) y así sucesivamente.

Original	א	ב	ג	ד	ה	ו	ז	ח	ט	י	כ	ל	מ	נ	ס	ע	פ	צ	ק	ר	ש	ת
Clave	ת	ש	ר	ק	צ	פ	ע	ס	נ	מ	ל	כ	י	ט	ח	ז	ו	ה	ד	ג	ב	א

Ejemplo con alfabeto en Español

Original	a	b	c	d	e	f	g	h	i	j	k	l	m	n	ñ	o	p	q	r	s	t	u	v	w	x	y	z
Clave	Z	Y	X	W	V	U	T	S	R	Q	P	O	Ñ	N	M	L	K	J	I	H	G	F	E	D	C	B	A

En todo caso, hay que tener presente que este método de cifrado se ideó para un abyad en el que solo se escriben las consonantes, que luego se vocalizan de manera más o menos arbitraria y, así, prácticamente cualquier palabra hebrea es pronunciable al cifrarse en atbash. En idiomas con escritura alfabética, como el español, es infrecuente que una palabra codificada en atbash sea pronunciable.

Cifrado Alberti

El cifrado de Alberti es el método de cifrado descrito por Leon Battista Alberti en su tratado 'De Cifris' en 1466. Constituye el primero cifrado por sustitución polialfabético conocido. El modo en el que se cambiaba de alfabeto no era periódico (a diferencia de otros cifrados posteriores como el de Vigenère. Para facilitar el proceso de cifrado/descifrado propone unos artilugios conocidos como 'discos de Alberti'.

Los discos de Alberti son artilugios que sirven de herramienta para realizar el cifrado de Alberti. Estos discos consisten en un armazón fijo en el que está grabado un alfabeto latín convencional ordenado y al final están las cifras 1, 2, 3 y 4. Unido a él por una pieza circular concéntrica y móvil con otro alfabeto grabado de forma que este círculo podía moverse con respecto al otro. De esta forma el usuario puede, mediante un giro del anillo móvil, emparejar el alfabeto del círculo de arriba con tantos alfabetos del círculo de abajo como giros distintos del anillo dé, hasta un máximo igual a los caracteres del alfabeto empleado.

El anillo fijo tiene 24 celdas iguales y en cada una de ellas hay grabada un símbolo del alfabeto latino en letras mayúsculas en el orden habitual. A continuación están los número 1, 2, 3, 4 por este orden. Este alfabeto se usará para el texto en claro

El anillo móvil tiene 24 celdas iguales y en cada una de ellas hay grabada un símbolo del alfabeto latino en minúscula o los símbolos '&', 'y', 'k' y 'h'. El orden de las celdas puede ser cualquiera. Este alfabeto se usará para el texto cifrado.

Por tanto el número de alfabeto distintos usables para un disco de Alberti determinado es de 24.

En el anillo fijo (el del alfabeto del texto en claro) aparecen las cifras del 1 al 4. Alberti aprovecha todas las combinaciones de de 2, 3 y 4 cifras de estos números (336=4^2+4^3+4^4 grupos) para poder establecer un código y así aumentar la seguridad del sistema
A este código se le llama código de recifrado (y en inglés superencipherment). Para aprovechar esta potencialidad tanto el receptor como el emisor deben compartir un libro de códigos que indique el significado del cada código usado. En este libro de códigos estarían aquellas palabras o frases de especial trascendencia en el ámbito de uso del cifrado, y por tanto a las que hay que dar mayor seguridad. Por ejemplo el libro de códigos podría atribuir al código '21' el significado 'Lanzar ataque' y al código '23' asignar el significado 'Replegarse'.
Por otro lado las cifras son introducidas para despistar y serán descartados cuando el receptor realice el descifrado. Por eso se dice que son caracteres nulos.
La clave del sistema viene definida por el orden de los símbolos en el anillo móvil y por la situación inicial relativa de los dos anillos. Para descifrar el receptor tiene que realizar operaciones inversas a las que realiza el que cifra. Para ello los giros que hay que ir realizando en el anillo móvil se indican en el texto cifrado siguiendo cierto convenio que el que descifra tiene que ir siguiendo. Hay 2 algoritmos para cifrar. Para explicarlos veamos un ejemplo de aplicación de ambos.

Ejemplo:
*Texto plano original: "**La guerra**"*
***Preprocesado**: Para adaptar el texto plano a las posibilidades de nuestro cifrador es necesario realizar un proceso de preprocesado. En este proceso convertimos al alfabeto del texto plano, aplicamos el código de recifrado (en este caso vamos a suponer que el libro de códigos no es aplicable) y si es necesario usamos caracteres nulos.*
En un primer paso pasamos obtenemos el texto plano: 'LAGVERRASIFARA' Vemos que la R doble puede producir problemas y entonces podemos considerar dos estrategias: eliminarla o introducir un carácter nulo. Con la primera estrategia obtendremos 'LAGVERASIFARA' y con la segunda 'LAGVER2RASIFARA'.
***Cifrado**: Vamos a usar uno de los dos textos obtenidos en el preprocesado para cifrar en cada uno de los algoritmos de cifrado.*
Primer método de cifrado
Texto a cifrar : 'LAGVER2RASIFARA'.
***Clave**: El orden del disco móvil es 'gklnprtuz&xysomqihfdbace'*
Se elige una letra del disco móvil como índice, únicamente conocido por el emisor y el receptor. Supongamos que es la 'g'.
Se hace coincidir la 'g' con la letra del disco móvil que queramos, por ejemplo la 'A'. Por tanto los discos quedan así:
ABCDEFGILMNOPQRSTVXZ1234 Anillo fijo
gklnprtuz&xysomqihfdbace Anillo móvil

El mensaje cifrado comenzará con la letra 'A' elegida para indicar cómo están los discos y se continúa sustituyendo hasta que se decide girar el disco. En ese momento se vuelve a poner la letra que coincide con la 'g' y vuelta a empezar. Por tanto si ciframos con la posición anterior de los discos hasta que pasamos a cifrar la letra 'S' y ahí cambiamos giramos y ponemos la 'g' en la 'Q' obtenemos el texto cifrado:
_LAGVER2RA_SIFARA texto a cifrar
AzgthpmamgQlfiyky texto cifrado

Observar que cuando realizó el giro los discos quedan en la posición relativa:
QRSTVXZ1234ABCDEFGILMNOP Anillo fijo
gklnprtuz&xysomqihfdbace Anillo móvil

Segundo método de cifrado
Texto a cifrar : 'LAGVERASIFARA'.
***Clave**: El orden del disco móvil es gklnprtuz&xysomqihfdbace*
Se elige una letra del disco fijo como índice, únicamente conocido por el emisor y el receptor. Supongamos que es la letra 'A'

Se hace coincidir la 'A' con la letra del disco móvil que queramos, por ejemplo la 'm'. Por tanto los discos quedan así:
ABCDEFGILMNOPQRSTVXZ1234 Anillo fijo
mqihfdbacegklnprtuz&xyso Anillo móvil
El mensaje cifrado comenzará con la letra 'm' elegida para indicar como están los discos y se continúa sustituyendo hasta que se decide girar el disco. En ese momento se cifra el número '3' que indicará al receptor que hay que mover el anillo. El anillo se moverá de tal forma que el resultado de cifrar el '3' sea alineado con nuestra clave.
Por tanto si queremos móvil el anillo móvil en la letra 'S' lo que tenemos que cifrar es 'LAGVERA3SIFARA'.
Por tanto el mensaje cifrado quedará:
_LAGVERA3SIFARA texto a cifrar
mcmbufpmsndhsls texto cifrado
Observar que cuando realizó el giro los discos quedan en la posición relativa:
ABCDEFGILMNOPQRSTVXZ1234 Anillo fijo
somqihfdbacegklnprtuz&xy Anillo móvil

Cifrado Vigenère

El **cifrado Vigenère** es un cifrado basado en diferentes series de caracteres o letras del **cifrado César** formando estos caracteres una tabla, llamada tabla de Vigenère, que se usa como clave. El cifrado de Vigenère es un cifrado de sustitución simple polialfabético.


Blaise de Vigenère


Giovan Battista Belasso

El cifrado Vigenère se ha reinventado muchas veces. El método original fue descrito por **Giovan Battista Belasso** en su libro de 1553 La cifra del Sig. **Giovan Battista Belasso**. Sin embargo, fue incorrectamente atribuido más tarde a **Blaise de Vigenère**, concretamente en el siglo XIX, y por ello aún se le conoce como el "cifrado Vigenère".
Este cifrado es conocido porque es fácil de entender e implementar, además parece irresoluble; esto le hizo valedor del apodo el código indescifrable (le chiffre indéchiffrable, en francés).

El cifrado Vigenère ganó una gran reputación por ser excepcionalmente robusto. Incluso el escritor y matemático Charles Lutwidge Dodgson (Lewis Carroll) dijo que el cifrado Vigenère era irrompible en el artículo "The Alphabet Cipher" para una revista de niños.

En 1917, "Scientific American" describió el cifrado Vigenère como imposible de romper. Esta reputación fue mantenida hasta que el **método Kasiski** resolvió el cifrado en el siglo XIX y algunos criptoanalistas habilidosos pudieron romper ocasionalmente el cifrado en el siglo XVI.
El cifrado Vigenère es lo suficientemente simple si se usa con discos de cifrado. Los Estados confederados de América, por ejemplo, utilizaron un disco de cifrado para implementar el cifrado Vigenère durante la Guerra Civil estadounidense. Los mensajes confederados fueron poco secretos, ya que los miembros de la Unión solían descifrar los mensajes.
Gilbert Vernam trató de arreglar el cifrado (creando el cifrado **Vernam-Vigenère** en 1918), pero no importa lo que hiciera, el cifrado sigue siendo vulnerable al criptoanálisis. (No confundir con el cifrado de Vernam)

```
mensaje:     P A R I S V A U T B I E N U N E M E S S E
clave:       L O U P L O U P L O U P L O U P L O U P L
criptograma:A O M X D K U K E P C T X J H T W S N I O
```

En este abecedario solo existen 27 letras donde A=0, B=1 , C=2 ... Z=26
En términos matemáticos puede expresarse como:

$$Yi = (Xi + Zi) mod T$$

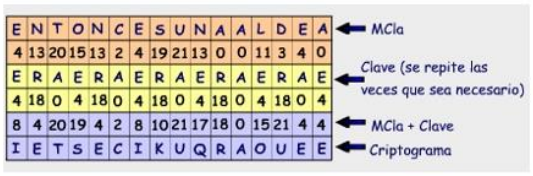
*Donde **Xi** es el número de ubicación de la letra, es decir, que P le corresponde al número 16 en modo horizontal y **Zi** la L en modo vertical le corresponde al número 11, y la letra **T** es el total de números del alfabeto.*

Entonces, para P y L la ecuación quedará de la siguiente manera: Yi= (16 + 11) mod 27. El resultado es: 0, donde 0 es igual a A en modo horizontal. Para A y O la ecuación quedará como: Yi = (0 + 15) mod 27. El resultado es 15, donde 15 es igual a O en modo horizontal. Para R y U la ecuación quedará como: Yi = (18 + 21) mod 27. El resultado es 12, donde 12 es igual a M en modo horizontal.
Otra manera es viendo la tabla, por ejemplo buscamos la P en la primera fila horizontal y la L en la primera columna vertical, en su intersección encontramos la letra A.

*Para descifrar hacemos la operación inversa, al resultado A le restamos la clave L y tenemos **Xi = (Yi - Zi) mod 27**, lo que nos da 16, correspondiente a la P que habíamos encriptado antes. Así hacemos con las demás letras.*
Para descifrar con la tabla, buscamos la clave en la primera fila horizontal, en este caso la L y bajamos verticalmente hasta encontrar la A del mensaje cifrado. De ahí nos vamos horizontalmente a la izquierda hasta la primera columna y tenemos la P que se había cifrado anteriormente.
con Xi = P, A, R, I y Zi = L, O, U, P, alternativamente, siendo T el número de letras del alfabeto.
Se observa que a una misma letra en el texto claro le pueden corresponder diferentes letras en el texto cifrado.

Resumiendo:

Es un algoritmo de desplazamiento que consiste en sustituir una letra por otra que se encuentra en una posición dada por la posición de la clave, por ejemplo cuando el mensaje es c, la letra "b" seria remplazada por la "d".



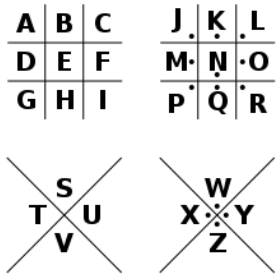
Cifrado: $Y_i = (X_i + Z_i) \bmod T$

Descifrado: $Y_i = (X_i - Z_i) \bmod T$

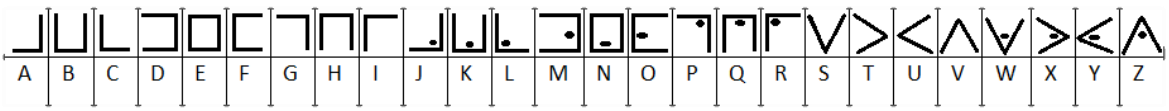
Y: es la posición del carácter ya cifrado.
i: representa el numero de carácter.
X: es la posición del carácter del mensaje.
Z: es la posición del carácter de la clave.
T: Total de caracteres en el alfabeto(en caso de usar el alfabeto ingles son: 27).

Cifrado Francmasón

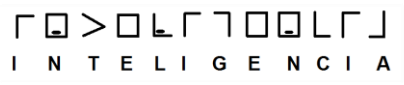
El cifrado francmasón es un cifrado por sustitución simple que cambia las letras por símbolos basándose en un diagrama. Sin embargo, el uso de símbolos no impide el criptoanálisis, y el criptoanálisis es idéntico al de otros métodos de cifrado por substitución simple. El método fue desarrollado por los francmasones a principios de los años 1700, debido a que era una sociedad secreta, era usado para mantener registros y para la correspondencia



El cifrado francmasón usa símbolos gráficos asignados en una clave como en este diagrama.

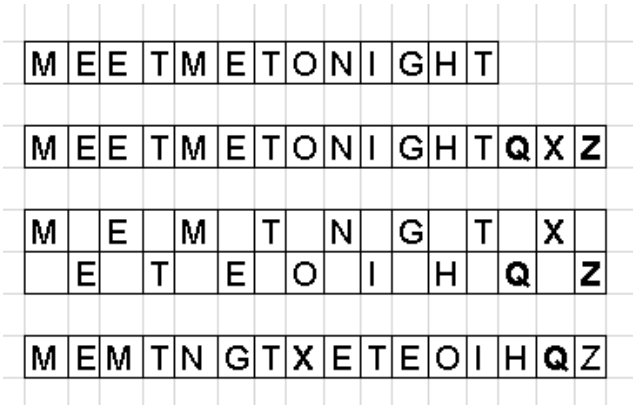


Ejemplo de codificación



Cifrado Rail Fence

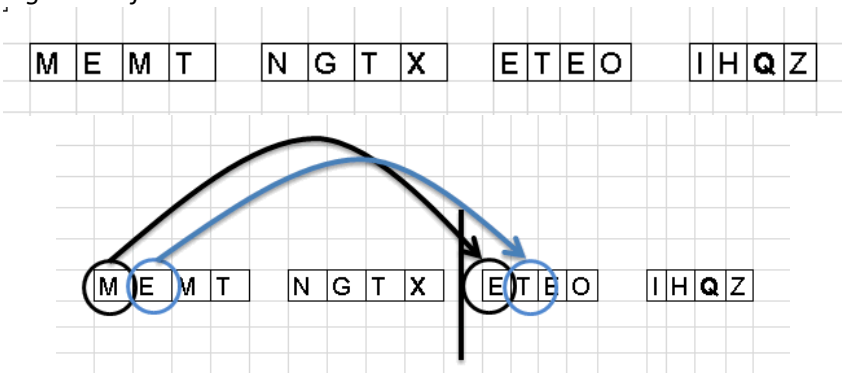
El Cifrado Rail Fence es una forma de cifrado por permutación (o transposición) que ha tomado por la forma en que se codifican los textos con él. En el cifrado rail fence, el texto plano se escribe hacia abajo diagonalmente a través de sucesivos raíles de una valla imaginaria, luego se escribe hacia arriba también diagonalmente. Dicho proceso se sigue hasta que se acaba el mensaje que se quiere cifrar. El mensaje cifrado se obtiene cogiendo los caracteres de las filas



Elemplo:
Texto Original: WE ARE DISCOVERED. FLEE AT ONCE'
W . . . E . . . C . . . R . . . L .
. . T . . . E
. E . R . D . S . O . E . E . F . E
. A . O . C .
. . A . . . I . . . V . . . D . . .
E . . . N . .
Resultado: WECRL TEERD SOEEF EAOCA IVDEN

El cifrado rail fence no es muy fuerte; el número de claves posibles es pequeño y un criptoanalista puede conseguir descifrar el mensaje a mano. La clave es el número de raíles.

Suponga que desea cifrar el siguiente mensaje en ingles: MEET ME TONIGHT. Se procede de la siguiente manera:
Se cuentan todas las letras
Si el numero es múltiplo de 4 ok. En otro caso, se agregan al final hasta que la cuenta lo sea. (En este caso 13 letras, agregamos QXZ, totalizando 16)
Se escribe el mensaje alternando caracteres una fila arriba y otra abajo como se ilustra
Se colocan ambas secuencias a continuación
Se divide el mensaje exactamente por la mitad
Se lee la primer letra de cada mitad, la segunda de la primer mitad, la segunda de la segunda mitad y tal
Se ignoran los agregados al final



Siguiendo los Pasos anteriores. Comenzamos a obtener el mensaje original. Siguiendo el gráfico anterior obtenemos: MEET…

Máquina Enigma

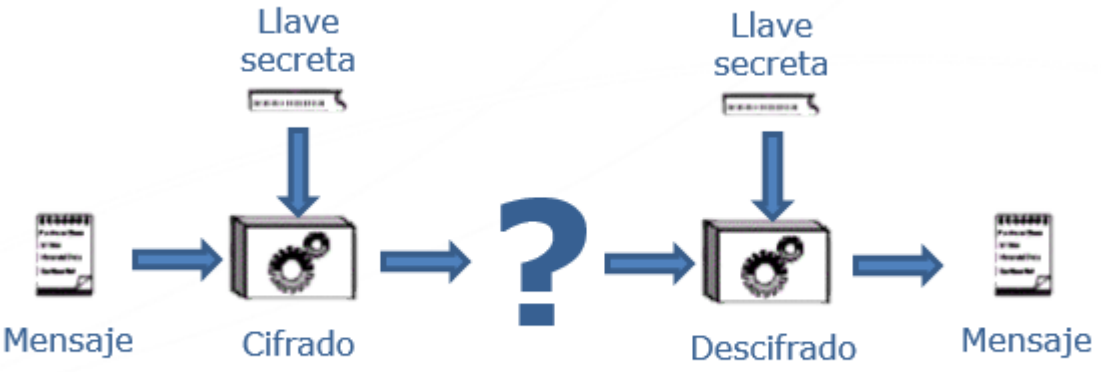
Enigma era el nombre de una máquina que disponía de un *mecanismo de cifrado rotatorio*, que permitía usarla tanto para cifrar como para descifrar mensajes. Varios de sus modelos fueron muy utilizados en Europa desde inicios de los años 1920.

Su fama se debe a haber sido adoptada por las fuerzas militares de Alemania desde 1930. Su facilidad de manejo y supuesta inviolabilidad fueron las principales razones para su amplio uso. Su sistema de cifrado fue finalmente descubierto y la lectura de la información que contenían los mensajes supuestamente protegidos es considerado, a veces, como la causa de haber podido concluir la Segunda Guerra Mundial al menos dos años antes de lo que hubiera acaecido sin su descifrado.

La máquina equivalente británica, **Typex**, y varias estadounidenses, como la **SIGABA** (o **M-135-C** en el ejército), eran similares a Enigma. La primera máquina moderna de cifrado rotatorio, de Edward Hebern, era considerablemente menos segura, hecho constatado por William F. Friedman cuando fue ofrecida al gobierno de Estados Unidos.

Criptografía simétrica

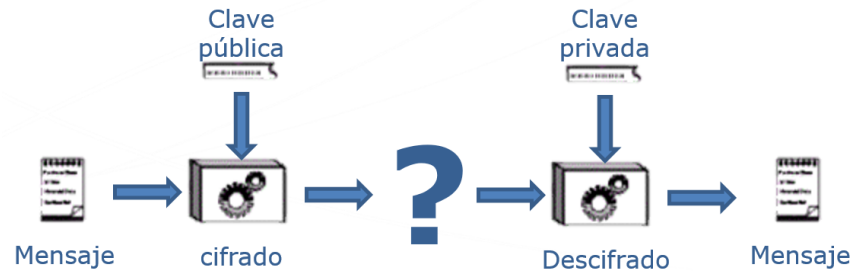
Los métodos criptográficos tradicionales operan a partir de una palabra o frase llave, que sirve para codificar y decodificar información, el conocido "password".
*Está llave **debe** ser conocida por los dos extremos de la comunicación, por lo que el punto débil de este método es justamente el proceso de difusión de la llave.*



Criptografía Asimétrica

Por el contrario, la Criptografía de Clave Pública asigna a cada extremo de la comunicación un par de llaves, una pública que cualquiera puede solicitar y conocer, y otra privada, cuya seguridad es fundamental para el éxito de la codificación.

Las llaves son una secuencia bastante larga de caracteres y números, generadas por un procedimiento matemático.
Para enviar un mensaje seguro a una persona, se codifica con la clave pública del destinatario. El sistema garantiza que el mensaje resultante sólo puede ser decodificado con la clave privada del destinatario (confidencialidad).
Como se tiene la seguridad de la identidad del destinatario gracias a su clave pública, nos aseguramos que el mensaje va al sitio correcto (autentificación).
***Clave privada**: será custodiada por su propietario y no se dará a conocer a ningún otro.*
***Clave pública**: será conocida por todos los usuarios.*
Esta pareja de claves es complementaria: lo que cifra una SÓLO lo puede descifrar la otra y viceversa.



Algoritmos Asimétricos conocidos
Diffie-Hellman
RSA
DSA
ElGamal
Criptografía de curva elíptica
Criptosistema de Merkle-Hellman
Goldwasser-Micali
Goldwasser-Micali-Rivest

Criptografía RSA

RSA es el primer y más utilizado algoritmo de este tipo y es válido tanto para cifrar como para firmar digitalmente.
La seguridad de RSA radica en el problema de la factorización de números enteros. Los mensajes enviados se representan mediante números, y el funcionamiento se basa en el producto, conocido, de dos números primos grandes elegidos al azar y mantenidos en secreto.

*Los mensajes enviados se representan **numéricamente**.*
*El funcionamiento se basa en el producto, conocido, de **dos números primos grandes** elegidos **aleatoriamente** en secreto.*
Estos números primos son del orden de 10 a la 200, y en aumento.
*En **RSA** cada usuario posee dos claves de cifrado: una pública y otra privada.*
El emisor busca la clave pública del receptor al momento de enviarle un mensaje, cifra este con esa clave.

1973 en GCHQ

Clifford Cocks, describe un método similar.
Pero no trasciende por la tecnología de la época

1977 en el MIT

Ron Rivest
Adi Shamir
Leonard Adleman

Una vez que el mensaje cifrado llega al receptor, este se ocupa de descifrarlo usando su clave privada.

Supongamos que Bernardo quiere enviar a Ana un mensaje secreto que solo ella pueda leer.
Ana envía a Bernardo una caja con una cerradura abierta, de la que solo Ana tiene la llave. Bernardo recibe la caja, escribe el mensaje, lo pone en la caja y la cierra con su cerradura *(ahora Bernardo no puede leer el mensaje)*. Bernardo envía la caja a Ana y ella la abre con su llave. En este ejemplo, la caja con la cerradura es la «***clave pública***» de Ana, y la llave de la cerradura es su «***clave privada***».

Técnicamente, Bernardo envía a Ana un «**mensaje llano**» **M** en forma de un número **m** menor que otro número **n**, mediante un protocolo reversible conocido como *padding scheme* («patrón de relleno»). A continuación genera el «***mensaje cifrado***» **c** mediante la siguiente operación:

$$c \equiv m^e \left(mod\ n\right)$$

donde **e** es la *clave pública* de Ana.
Ahora Ana descifra el mensaje en clave **c** mediante la operación inversa dada por

$$m \equiv c^d \left(mod\ n\right)$$

donde **d** es la *clave privada* que solo Ana conoce.

Generación de las claves:

1. Cada usuario i del sistema elige una pareja de números primos $\mathrm{p_i}, \mathrm{q_i}$.
Por motivos de seguridad, estos números deben escogerse de forma aleatoria y deben tener una longitud en bits parecida. Se pueden hallar primos fácilmente mediante el ***test de primalidad****.*
2. Se calcula $\mathrm{n_i} = \mathrm{p_i}.\mathrm{q_i}$ *(n se usa como el módulo para ambas claves, pública y privada.)*
3. Se calcula la función de Euler $\varphi(\mathrm{n_i}) = (\mathrm{p_i} - 1)(\mathrm{q_i} - 1)$.
4. Se elige arbitrariamente $\mathrm{e_i}$, que cumpla con $0 < \mathrm{e_i} < \varphi(\mathrm{n_i})$, y sea coprimo con $\varphi(\mathrm{n_i})$.
   -$\mathrm{e_i}$ se da a conocer como el exponente de la clave pública.
   -Si se escoge un $\mathrm{e_i}$ con una suma encadenada corta, el cifrado será más efectivo. Un exponente $\mathrm{e_i}$ muy pequeño (p. ej. $\mathrm{e_i}$ =3) podría suponer un riesgo para la seguridad.
5. Se determina un $\mathrm{d_i}$(mediante aritmética modular) que satisfaga la congruencia $\mathrm{e}.d \equiv 1\left(mod\ \varphi(\mathrm{n_i})\right)$, es decir, que $\mathrm{d_i}$ sea el multiplicador modular inverso de $e_\mathrm{i}\ mod\ \varphi(\mathrm{n_i})$
   -expresado de otra manera,$d_\mathrm{i}e_\mathrm{i} - 1$ es dividido exactamente por $\varphi(\mathrm{n_i}) = (\mathrm{p_i} - 1)(\mathrm{q_i} - 1)$.
   -esto suele calcularse mediante el ***algoritmo de Euclides*** extendido.
   -d_i se guarda como el exponente de la clave privada.

La clave pública es $(\mathrm{n_i}, \mathrm{e_i})$, esto es, el módulo y el exponente de cifrado. La clave privada es $(\mathrm{n_i}, \mathrm{d_i})$, esto es, el módulo y el exponente de descifrado, que debe mantenerse en secreto.

Cifrado: Ana comunica su clave pública (n_i, e_i) a Bernardo y guarda la clave privada en secreto. Ahora Bernardo desea enviar un mensaje **M** a Ana.
Primero, Bernardo convierte **M** en un número entero **m** menor que **n** mediante un protocolo reversible acordado de antemano. Luego calcula el texto cifrado **c** mediante la operación $c \equiv m^{e_i} (mod\ n_i)$
Esto puede hacerse rápido mediante el método de exponenciación binaria. Ahora Bernardo transmite **c** a Ana.
Descifrado: Alicia puede recuperar **m** a partir de **c** usando su exponente **d** de la clave privada mediante el siguiente cálculo: $m \equiv c^{d_i} (mod\ n_i)$
Ahora que tiene **m** en su poder, puede recuperar el mensaje original **M** invirtiendo el *padding scheme*.
El procedimiento anterior funciona porque: $\boldsymbol{c^{d_i} = (m^e)^d \equiv m^{ed} (mod\ n)}$
Esto es así porque, como hemos elegido d y e de forma que $ed = 1 + k\varphi(n)$, se cumple que $m^{ed} = m^{1+k\varphi(n)} = m(m^{\varphi(n)})^k \equiv m(mod\ n)$.
La última congruencia se sigue directamente del ***teorema de Euler*** cuando **m** es *coprimo* con **n**. Puede demostrarse que las ecuaciones se cumplen para todo **m** usando congruencias y teorema chino del resto. Esto muestra que se obtiene el mensaje original:
$m \equiv c^d (mod\ n)$.

Podemos reconocer la importancia de la criptografía en la informática, quedado claro lo más básico de la criptografía computacional. En este tiempo es considerada una parte imprescindible en cualquier sistema y pienso lo será aún más; si cabe pensar que en el futuro prevé que todo fluya en la red incluso, lo más confidencial.
He aquí la importancia del cifrado, la autentificación, la comprobación de integridad y los posibles usos que de la criptografía se pueda sacar.
Con la mirada atenta en el futuro (quizás la criptografía cuántica) y confiando en la seguridad que nos proporcionan los sistemas actuales, la criptografía nos ayuda a cumplir los principios con los que la informática nació: ***el tratamiento seguro de información***

Números primos

- *Un entero positivo p mayor que 1 se llama primo si los únicos divisores de p son 1 y p*
- *Un entero positivo mayor que 1 que no es primo se denomina compuesto*

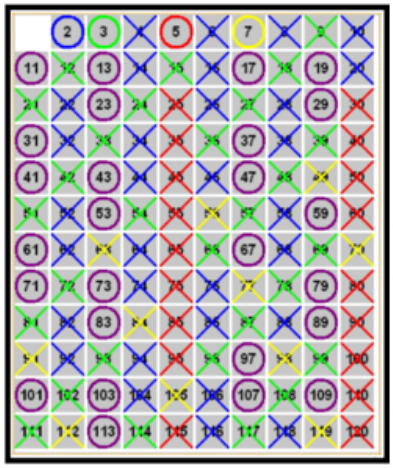
#	n	Fecha del descubrimiento	Descubridor
1	2	*antigüedad*	*desconocido*
2	3	*antigüedad*	*desconocido*
3	5	*antigüedad*	*desconocido*
4	7	*antigüedad*	*desconocido*
5	13	1456	*anónimo*
6	17	1588	Cataldi
7	19	1588	Cataldi
8	31	1772	Euler
9	61	1883	Pervushin
10	89	1911	Powers
11	107	1914	Powers
12	127	1876	Lucas
13	521	30-01-1952	Robinson
14	607	30-01-1952	Robinson
15	1.279	25-06-1952	Robinson
16	2.203	07-10-1952	Robinson

Número primo	Fecha de descubrimiento
$2^{42643801}-1$	2009
$2^{37156667}-1$	2008
$2^{32582657}-1$	2006
$2^{30402457}-1$	2005

Eratostenes

- *Matemático, astrónomo y geógrafo griego*
- *Hizo contribuciones acerca de las dimensiones de la tierra*
- *Compañero de Arquímedes*

Criba de Eratóstenes: Es un método para hallar todos los números primos menores que un natural N dado



Algoritmo Criba de Eratóstenes (Complejidad $\mathcal{O}(n \log^2 n \log\log n)$)

Entrada: Un número natural n

Salida: El conjunto de números primos anteriores a n (incluyendo n)

1. Escriba todos los números naturales desde 2 hasta n
2. **Para** i desde 2 hasta $\lfloor \sqrt{n} \rfloor$ **haga lo siguiente:**
    1. **Si** i no ha sido marcado **entonces:**
        1. **Para** j desde i hasta $n \div i$ **haga lo siguiente:**
            1. Ponga una marca en $i \times j$
3. **El resultado es:** Todos los números sin marca

Programación dinámica

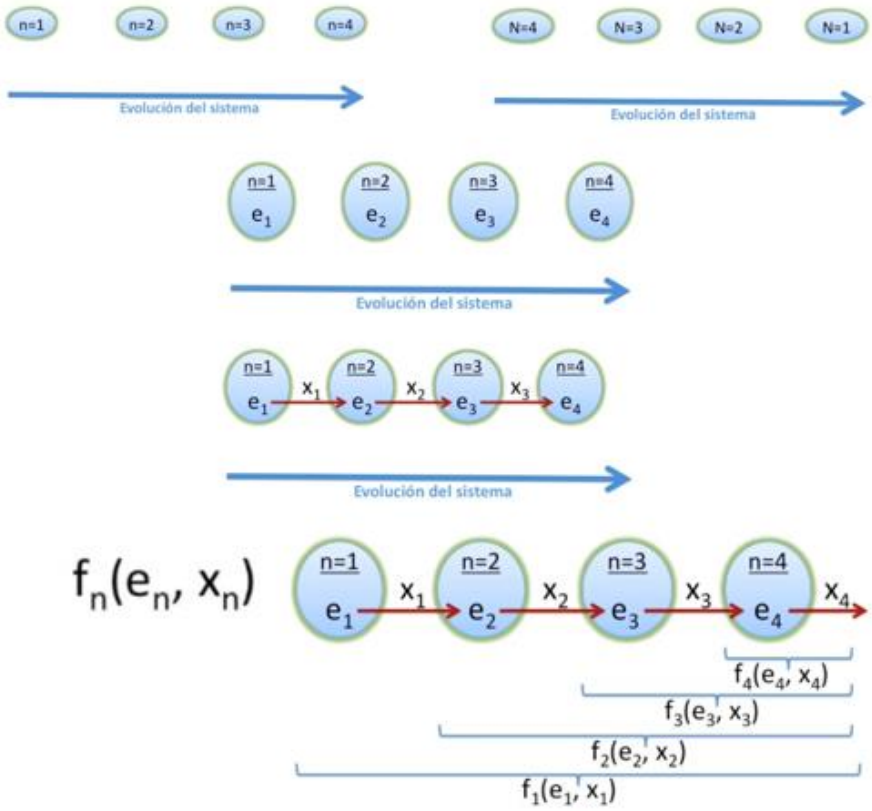
La programación dinámica es un método para reducir el tiempo de ejecución de un algoritmo mediante la utilización de sub-problemas superpuestos y subestructuras óptimas, como se describe a continuación. El matemático Richard Bellman inventó la programación dinámica en 1953 que se utiliza para optimizar problemas complejos que pueden ser discretizados y secuencializados.

La programación dinámica resuelve los subprogramas una sola vez y guarda sus soluciones en una tabla para su futura utilización; basándose en que “en una secuencia de decisiones óptima toda subsecuencia ha de ser también óptima”.

Para que un problema pueda ser abordado por esta técnica ha de cumplir dos condiciones:

- La solución al problema ha de ser alcanzada a través de una secuencia de decisiones, una en cada etapa.
- Dicha secuencia de decisiones ha de cumplir el principio de óptimo.

Técnica cuantitativa que permite encontrar las decisiones óptimas en un proceso dividido en fases
Elementos que definen un problema de programación dinámica son:
***Etapas *Estados *Variables de decisión *Función recurrente o recursiva**
Etapas: Son el periodo de tiempo, el lugar, el contexto, la fase o situación en donde se produce un cambio debido a una decision
Estados: Los estados muestran la situación actual del sistema cuando nos encontramos en la etapa n.
Variables de decisión: hacen referencia a toma de decisiones que se produce en una etapa y que provoca un cambio en el estado actual del sistema.
Función recurrente: La función de recurrencia refleja el comportamiento del sistema en función de los estados y las variables de decisión.



La programación dinámica es utilizada en compiladores, que consiste en solucionar cierto problema dividiéndolo en subproblemas más sencillos, calculando sus resultados y almacenándolos. Estos resultados posteriormente se utilizan para la resolución del problema final.
Almacenar resultados de subproblemas es una gran ventaja en cálculos dónde se repiten las mismas operación múltiples veces, mediante el método de la programación dinámica estas operaciones sólo se realizan una vez y se guarda la solución.
Se dice de la programación dinámica que es un método para resolver problemas que exhiben propiedades de problemas sobrepuestos y estructura óptima.

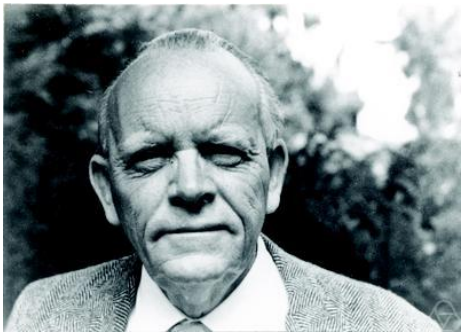
En grandes líneas, el diseño de un algoritmo de Programación Dinámica consta de los siguientes pasos:

1. ***Planteamiento de la solución como una sucesión de decisiones y verificación de que ésta cumple el principio de óptimo.***
2. ***Definición recursiva de la solución.***
3. ***Cálculo del valor de la solución óptima mediante una tabla en donde se almacenan soluciones a problemas parciales para reutilizar los cálculos.***
4. ***Construcción de la solución óptima haciendo uso de la información contenida en la tabla anterior.***

En general, los algoritmos obtenidos mediante la aplicación de esta técnica consiguen tener complejidades (espacio y tiempo) bastante razonables, pero debemos evitar que el tratar de obtener una **complejidad temporal** de orden polinómico conduzca a una **complejidad espacial** demasiado elevada
La Programación Dinámica se adapta a problemas de carácter secuencial, por ejemplo:

- Búsqueda de caminos más cortos entre dos puntos
- Planificación de tareas
- Gestión de recursos
- Gestión de stock

Algoritmo de vuelta atrás (BackTracking)



Vuelta atrás, (Backtracking) es una estrategia para encontrar soluciones a problemas que satisfacen restricciones. El término "backtrack" fue acuñado por primera vez por el matemático estadounidense D. H. Lehmer en la década de 1950.
El retroceso o vuelta atrás es una técnica de resolución de problemas que ***realiza una búsqueda exhaustiva, sistemática y organizada sobre el espacio de búsqueda del problema***, aplicable a problemas de optimización, juegos, búsquedas entre otros.
Se llaman algoritmos de vuelta atrás, porque en el caso de no encontrar una solución a una sub-tarea se retrocede a la sub-tarea anterior y se prueba otro camino diferente a los probados anteriormente.

Los problemas que manejan los algoritmos exhaustivos, se caracterizan por:

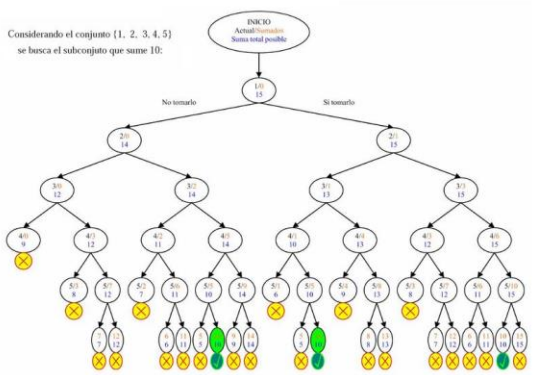
a. **Se trata generalmente de problemas de optimización, con o sin restricciones.**
b. **La solución es expresable en forma de una secuencia de decisiones**.
c. **Existe una función denominada factible que permite averiguar si en una secuencia de decisiones, la solución en curso, viola o no las restricciones.**
d. **Existe una función, denominada solución, que permite determinar si una secuencia de decisiones factible es solución al problema planteado.**

El esquema general de solución presenta los siguientes pasos:

a. **Vuelta Atrás hace un recorrido en profundidad del espacio de búsqueda partiendo de la raíz.**

b. **El recorrido en profundidad regresa sobre sus pasos, retrocede, cada vez que encuentra un camino que se ha acabado o por el que no puede continuar.**

c. **En un recorrido en profundidad o en anchura sobre un espacio de búsqueda se conoce de antemano el orden en que se van a generar o recorrer, sus nodos**.



Este tipo de metodología es muy genérica, ya que trata de recorrer ***todas las posibles soluciones*** y ***encontrar la más apropiada*** para cada caso o las soluciones que satisfacen una condición, con lo que puede ser aplicada en la mayoría de problemas. Por ello, esta técnica de resolución es usada en muchos ámbitos de la programación, por ejemplo, para el cálculo de expresiones regulares o para tareas de reconocimiento de texto y de sintaxis de lenguajes regulares; también esta técnica es usada en diversos lenguajes de programación tales como ***Planner y Prolog***. Además, se usa en los análisis sintácticos de los compiladores. Su uso en inteligencia artificial ha sido muy importante, dando lugar a nuevos tipos de búsquedas como el A estrella.

Ejemplos de problemas comunes resueltos usando Vuelta Atrás

* Sudoku
* Problema de los movimientos de un caballo
* Las ocho reinas

Esteganografía (Apéndice a criptografía)

(No está en programa y no va en los parciales)

Esta ciencia ha suscitado mucho interés en los últimos años debido a que ha sido utilizada por organizaciones criminales y terroristas. No obstante, no se trata de algún nuevo ingenio, se lleva empleando desde la antigüedad más remota.

Esta presentación pretende introducirlos en el campo de la esteganografía, clarificando sus diferencias con la criptografía y mostrando ejemplos de software para hacer uso de esta técnica.


USA TODAY
Home
News
Money
Sports
Life
Tech
Main Categories
Tech briefs
Web Guide
Tech Investor
Product reviews

CyberSpeak

• E-mail this story • Subscribe to the newspaper • Sign-up for e-mail news

12/19/2001 - Updated 09:30 AM ET

Bin Laden's messages could be hiding in plain sight

Now we have to worry about steganography. That's right: steganography. No not the ancient art of writing on a stegosaurus. It's a way to conceal a coder

Uso extendido

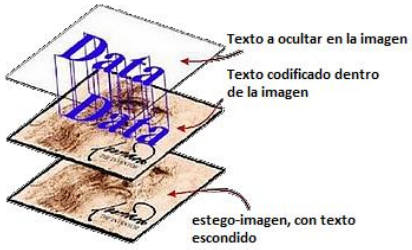
ETA no es la única organización terrorista que acude a programas de encriptación para proteger sus documentos y bases de datos.

Al Qaeda tiene incluso una aplicación propia y de distribución libre, el conocido como **Mujahidin's Secret**, capaz de codificar y enviar de forma segura todo tipo de archivos a través de internet.

También se ha detectado el uso de la esteganografía, una técnica muy antigua pero que ahora aprovecha las nuevas tecnologías y que permite camuflar información relevante en documentos, como fotografías, textos o incluso canciones, en apariencia inofensivos.

Recortes de periódico: "La esteganografía no es sólo un ingenio teórico"

Del griego *steganos* (oculto) y *graphos* (escritura), la esteganografía se puede definir como la ocultación de información en un canal encubierto con el propósito de prevenir la detección de un mensaje oculto.



La esteganografía estudia técnicas cuyo fin es insertar información sensible dentro de otro archivo. A este archivo se le denomina archivo contenedor (gráficos, documentos, programas ejecutables, etc.). De esta forma, se consigue que la información pase inadvertida a terceros, de tal forma que sólo sea recuperada por un usuario legítimo que conozca un determinado algoritmo de extracción de la misma.

Aunque κρσπτ ός (criptos, oculto) y steganoz (steganos, encubierto) puedan parecer en un principio términos equivalentes, o al menos similares, son cosas completamente distintas.

La criptografía es el arte de escribir de forma enigmática (según la Real Academia Española), mientras que la Esteganografía es el arte de escribir de forma oculta. Puede que sigan pareciendo similares, pero las connotaciones toman mucho valor al analizarlo detenidamente: **la criptografía tiene su fuerza en la imposibilidad de comprender el mensaje, mientras que la Esteganografía la tiene en el desconocimiento que el mensaje siquiera existe.**

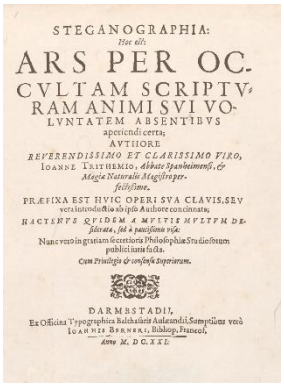
Aplicado al campo informático, podemos dar los siguientes ejemplos: nosotros podríamos robar un mensaje cifrado con relativa facilidad, pero aún sabiendo que contiene información importante seríamos incapaces de obtener información alguna de él (así la criptografía ha cumplido con su cometido). Respecto a la Esteganografía, nosotros podríamos capturar el tráfico completo de un individuo y tratar de analizarlo completamente (y el "ruido de fondo" hoy en día es mucho), sin tener la certeza de que haya o no un mensaje oculto. **La Esteganografía NO es un tipo de criptografía: son técnicas distintas e independientes, si bien pueden complementarse entre ellas (y de hecho lo suelen hacer).**

Historia

Más de 400 años antes de Cristo, Herodoto ya reflejó en su libro “Las Historias” el uso de la esteganografía en la antigua Grecia. En dicho libro describe como un personaje toma un cuadernillo de dos hojas o tablillas; raya bien la cera que las cubre y en la madera misma graba un mensaje y lo vuelve a cubrir con cera.
Otra historia, en el mismo libro, describe como otro personaje rasura a navaja la cabeza de uno de sus esclavos y le tatúa un mensaje en el cuero cabelludo. Así, espera a que le vuelva a crecer el cabello y lo manda al receptor del mensaje con instrucciones de que le rasuren la cabeza.

Tablilla para escribir con mensaje oculto grabado en la madera bajo la cera

Un ejemplo histórico más de uso de la esteganografía es el libro “Hypnerotomachia Poliphili” de Francesco Colonna, que data de 1499. En él, tomando la primera letra de sus 38 capítulos se puede leer “Poliam frater Franciscus Columna peramavit”, que se traduce por “El hermano Francesco Colonna ama apasionadamente a Polia”.


STEGANOGRAPHIA:
Hoc eſt:
ARS PER OC-
CVLTAM SCRIPTV-
RAM ANIMI SVI VO-
LVNTATEM ABSENTIBVS
aperiendi certa;
AVTHORE
REVERENDISSIMO ET CLARISSIMO VIRO,
IOANNE TRITHEMIO, Abbate Spanheimenſi, &
Magiæ Naturalis Magiſtro per-
fectiſſimo.
PRÆFIXA EST HVIC OPERI SVA CLAVIS, SEV
vera introductio ab ipſo Authore concinnata,
HACTENVS QVIDEM A MVLTIS MVLTVM DE-
ſiderata, ſed à pauciſſimis viſa:
Nunc verò in gratiam ſecretioris Philoſophiæ Studioſorum
publici iuris facta.
Cum Priuilegio & conſenſu Superiorum.
DARMSTADII,
Ex Officina Typographica Balthaſaris Aulændij, Sumptibus verò
IOANNIS BERNERI, Bibliop. Francof.
Anno M. DC. XXI.

Los trabajos de Johannes Trithemius son considerados como influyentes en la evolución de la criptografía y la esteganografía. Define sistemas para ocultar mensajes secretos dentro de mensajes aparentemente inofensivos.
El libro “Steganographia”, que tiene una apariencia de texto mágico o de brujería, tuvo una circulación reducida en el siglo XVI, concretamente en círculos privados, hasta que se publicó finalmente en 1606. En general, se traduce del término inglés “steganography”, que a su vez proviene del título del libro “Steganographia”, escrito por el abad alemán Johannes Trithemius (1462-1516) en 1499.

La técnica de micropuntos fue inventada por los alemanes durante la Segunda Guerra Mundial y fue usada de forma muy activa durante la época de la Guerra Fría. La técnica se basa en esconder puntos minúsculos en fotografías, tan pequeños que para el ojo humano e incluso para instrumentos ópticos básicos como lupas, que resultan invisibles, pero que forman un patrón de información significativa.

Debido a la naturaleza analógica de esta técnica, resultaba fácilmente detectable para los servicios de inteligencia, si bien advertir la presencia de mensajes esteganografiados no siempre significa que puedan ser legibles. Aún así, descubrir la presencia de un mensaje esteganografiado se considera un fracaso de la Esteganografía que lo soporta, pues la imposibilidad de comprender su contenido conforma su capa de cifrado.
Más familiar resulta el ejemplo de tinta invisible. Son muchos los niños que juegan a enviarse mensajes escritos con zumo de limón o substancias similares (con alto contenido en carbono), de tal forma que al

calentar la superficie sobre la que se escribe el mensaje, éste aparece en un tono color café. Esta técnica se puede hacer más compleja si se involucran reacciones químicas.

Queda patente que la esteganografía ha estado presente en nuestra civilización desde tiempos inmemoriales y ha sido tradicionalmente empleada por las agencias militares y de inteligencia, los criminales y la policía, así como por civiles que desean saltarse restricciones gubernamentales. Ahora bien, mientras la esteganografía clásica se basaba únicamente en el desconocimiento del canal encubierto bajo uso, en la era moderna se emplean canales digitales (imagen, video, audio, protocolos de comunicaciones, etc.) para alcanzar el objetivo. En muchos casos el objeto contenedor es conocido, lo que se ignora es el algoritmo de inserción de la información en dicho objeto.

Definiciones y fundamentos teóricos

La esteganografía es una solución al clásico problema del prisionero. En una prisión de alta seguridad dos internos en celdas separadas, Rómulo y Remo, se quieren comunicar para elaborar un plan de fuga. Ahora bien, toda comunicación intercambiada entre ellos es examinada por un guardia que los aísla por completo ante cualquier sospecha de comunicación encubierta. Con la esteganografía el guardia inspecciona mensajes aparentemente inocuos que contienen un canal subliminal muy útil para los prisioneros.
Se pueden observar distintos actores implicados en el campo de la esteganografía:

- **Objeto contenedor**: se trata de la entidad que se emplea para portar el mensaje oculto. Acudiendo al ejemplo de los mensajes sobre el cuero cabelludo, el objeto contenedor es el esclavo en sí.
- **Estego-objeto**: se trata del objeto contenedor más el mensaje encubierto. Siguiendo con el ejemplo, se trata del esclavo una vez se ha escrito en su cuero cabelludo el mensaje y se le ha dejado crecer el pelo.
- **Adversario**: son todos aquellos entes a los que se trata de ocultar la información encubierta. En el ejemplo de la prisión, se trata del guardia que entrega los mensajes a uno y otro prisionero. Este adversario puede ser pasivo o activo. Un adversario pasivo sospecha que se puede estar produciendo una comunicación encubierta y trata de descubrir el algoritmo que se extrae del estego-objeto, pero no trata de modificar dicho objeto. Un adversario activo, además de tratar de hallar el algoritmo de comunicación encubierta, modifica el estego-objeto con el fin de corromper cualquier intento de mensajería subliminal.
- **Estegoanálisis**: ciencia que estudia la detección (ataques pasivos) y/o anulación (ataques activos) de información oculta en distintas tapaderas, así como la posibilidad de localizar la información útil dentro de la misma (existencia y tamaño).

- **La invisibilidad perceptiva**. se refiere al grado en que la información oculta incluida debe pasar inadvertida a los sentidos de todo el mundo menos al destinatario de la misma. En la mayoría de los casos se quiere alcanzar la mayor invisibilidad perceptiva posible, pero hay algunas excepciones, por ejemplo las marcas de agua de los billetes, que son visibles al trasluz.
- **La invisibilidad estadística o algorítmica**, se refiere al grado en que la información oculta es invisible ante estegoanálisis.

- **La robustez**, de la información oculta, es la resistencia que tienen ante la manipulación inocente de la imagen, audio, etc. que porta la información oculta. Por ejemplo resistencia ante la compresión, que alguien pueda realizar sin intención de borrar la información oculta.
- **La seguridad**, de un método de esteganografía es la robustez de la información oculta ante ataques intencionados. Mide el grado de dificultad de eliminación o extracción de la información oculta, para un atacante que crea que dicha información oculta existe, pero no disponga de la clave que se utilizó para ocultarla.
- **La capacidad**, mide la cantidad de información oculta que se puede incluir por cantidad de información portadora sin incumplir ninguno de los demás requisitos.
- **La forma de detección**, puede ser de dos tipos: detección ciega y detección informada. En el caso de la detección ciega, sólo se dispone de la clave que se utilizó para incluir la información oculta, mientras que en la detección informada, se dispone de la información portadora original. La detección informada tiene menor probabilidad de error.
- **La complejidad computacional**, a veces es requisito que sea muy baja, por ejemplo, en el caso de aplicaciones que transmitan música en tiempo real y que a su vez incluyan información oculta en la misma. En otras aplicaciones, que no tengan requisitos de tiempo real, quizá puedan permitirse mayor complejidad. Pero en general, es deseable que sea baja.

Esteganografía usando imágenes como portadoras

Esta presentación se va a centrar en el objeto contenedor más utilizado: las imágenes digitales. Especialmente, en formato BMP por su sencillez (es un formato de fichero sin compresión). Las ideas presentadas se pueden extender a otros formatos (JPG, PNG, etc.) y a otros contenedores (vídeos, documentos, etc.) siempre que se respeten las particularidades de cada formato.

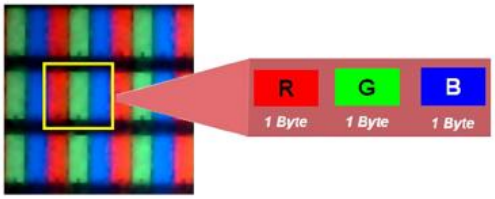
Existen 3 técnicas para el uso de imágenes en general

- **Sustitución de bits del objeto contenedor**
- **Inserción de bits en el objeto contenedor**
- **Creación de una imagen contenedor partiendo de la información a ocultar**

Sustitución de bits del objeto contenedor

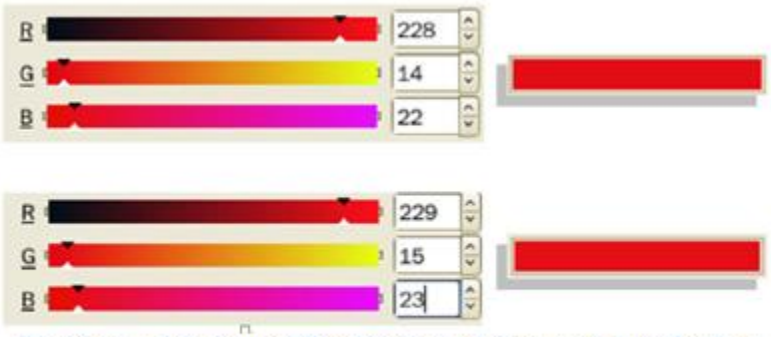
Esta técnica consiste en sustituir ciertos bits del fichero contenedor por los de la información a ocultar. La ventaja de este enfoque es que el tamaño del fichero contenedor no se ve alterado y, gracias a la redundancia y/o exceso de detalle en dichos ficheros, en muchas ocasiones tampoco su calidad.
Por ejemplo, en un fichero de sonido se pueden emplear los bits que no son audibles por el oído humano para ser reemplazados por los bits del mensaje.
Si se trabaja con imágenes, el método tradicional consiste en sustituir los bits menos significativos (LSB), en una escala de color de 24 bits (mas de 16 millones de colores). Esto se traduce tan sólo en que un píxel con un tono rojo se ve un 1% más oscuro. En muchos casos son cambios inapreciables a los sentidos

humanos que tan sólo pueden ser detectados mediante análisis computacional de la estructura de los ficheros.
Los archivos BMP son un formato estándar de imagen de mapa de bits en sistemas operativos DOS, Windows y válido para MAC y PC. Soporta imágenes de 24 bits (millones de colores) y 8 bits (256 colores), y puede trabajar en escala de grises, RGB y CMYK.



Estructura ilustrativa de píxel de una imagen

Cada píxel de un archivo BMP de 24 bits está representado por tres bytes. Cada uno de estos bytes contiene la intensidad de color rojo, verde y azul (RGB: red, green, blue). Combinando los valores en esas posiciones podemos obtener los 224, más de 16 millones, de colores que puede mostrar un píxel.
A su vez, cada byte contiene un valor entre 0 y 255, o lo que es lo mismo, entre 00000000 y 11111111 en binario, siendo el dígito de la izquierda el de mayor peso. Lo que demuestra que se pueden modificar los bits menos significativos de un píxel sin producir mayor alteración.



Efecto visual de la modificación de los bits menos significativos de las componentes RGB de un píxel

Se ha aumentado en una unidad cada componente RGB del píxel y el efecto es inapreciable para el ojo humano. De hecho, si se tiene en cuenta que un píxel esta rodeado por otros, el efecto visual si no se modifica su entorno pasa aun más inadvertido.
La implicación es que, utilizando cambios de un bit en cada componente de un píxel, se puede encajar tres bits de información oculta por píxel sin producir cambios notables en la imagen.

Esto se puede hacer para cada píxel de una imagen. Se necesitan ocho píxeles para ocultar tres bytes de información, en codificación ASCII esto son 3 letras de información oculta. Así, en una imagen BMP de 502x126 píxeles se puede ocultar un mensaje de 23.719 caracteres ASCII.

Imagen en la que se ha ocultado información en los bits menos significativos de sus píxeles

Para el caso de imágenes BMP la esteganografía por sustitución es bastante sencilla, la técnica se complica cuando se trata con otros formatos, pero la idea básica es la misma:
• Modificación de los índices que apuntan a la paleta de colores en un fichero GIF.
• Sustitución de coeficientes cuantificados DCTs en archivos JPG.

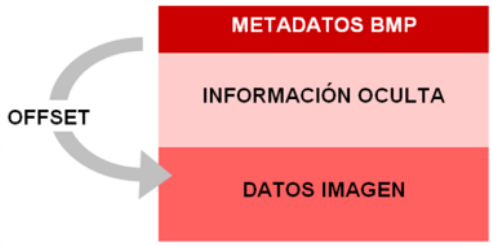
Esta técnica tiene un fallo latente de concepto: asume que la información almacenada originalmente en los bits menos significativos es aleatoria, con lo que su modificación para introducir información oculta no devela que la imagen está tratada. Esto no es cierto y puede servir como base para un mecanismo de estegoanálisis que se explica más adelante.

2- Inserción de bits en el objeto contenedor

En este caso se añaden los bits de información a partir de una determinada marca estructural del fichero (fin de fichero o EOF, espacios de padding o alineamiento, etc.). Esta opción presenta el inconveniente que sí se modifica el tamaño del objeto contenedor, con lo cual se puede levantar sospechas.
Para extrapolar esta idea al ejemplo de las imágenes BMP hay que comprender primero cómo se estructura dicho formato. Los primeros 54 bytes contienen los metadatos de la imagen, que se dividen de la siguiente manera:
•2 bytes: Contienen siempre la cadena 'BM', que revela que se trata de un BMP.
•4 bytes: Tamaño del archivo en bytes.
•4 bytes: reservados (para usos futuros), contienen ceros.
•4 bytes: offset, distancia entre cabecera y primer píxel de la imagen.
•4 bytes: tamaño de los metadatos (esta estructura en sí).
•4 bytes: ancho (número de píxeles horizontales).
•4 bytes: alto (número de píxeles verticales).
•2 bytes: número de planos de color.
•2 bytes: profundidad de color.
•4 bytes: tipo de compresión (vale cero, ya que BMP es un formato no comprimido).
•4 bytes: tamaño de la estructura imagen.
•4 bytes: píxeles por metro horizontal.
•4 bytes: píxeles por metro vertical.
•4 bytes: cantidad de colores usados.

•4 bytes: cantidad de colores importantes.

Dada esta estructura, la forma trivial de ocultar datos consiste en ocultarlos justo después de los metadatos (entre los metadatos y los datos de la imagen en sí) y modificar el campo offset (distancia entre los metadatos y los píxeles de la imagen). De esta forma se puede dejar espacio para todo el contenido adicional que se desee albergar.



Esquema del resultado de esteganografía por inserción en imágenes BMP

La ilustración pone de manifiesto que esta técnica no es muy sigilosa. Si los datos a ocultar son considerablemente pesados (varios megabytes); es un tanto sospechoso tener un icono de 10x10 píxeles que ocupe 5 megabytes. Así, la persona encargada de ocultar la información debe repartirla sobre diferentes imágenes si desea que el cambio no sea tan notorio.

3- Creación de una imagen contenedor partiendo de la información a ocultar

Esta alternativa consiste simplemente en generar un archivo contenedor con la propia información a ocultar, en lugar de obtener el archivo contenedor por separado y manipularlo para incluir dicha información.

Por ejemplo, dado un algoritmo específico de reordenamiento de los bytes de los datos a ocultar se puede generar una secuencia de píxeles de una imagen que tengan cierto significado visual. Si el receptor conoce el algoritmo de reordenamiento, la transmisión de información es posible.

Otros potadores de información

***Documentos de texto como portadores**: La esteganografía sobre texto es poco eficiente, debido a que casi cualquier cambio llama la atención y, con los métodos que no se llama la atención tienen una baja capacidad subliminal.

-Inserción de blancos.
-Métodos sintácticos.
-Métodos semánticos.
-Creación de un texto contenedor partiendo de la información a ocultar.

***Audios como portadores**:

-Low bit encoding. -Spread Spectrum.
-Echo data hiding.

Estegoanálisis

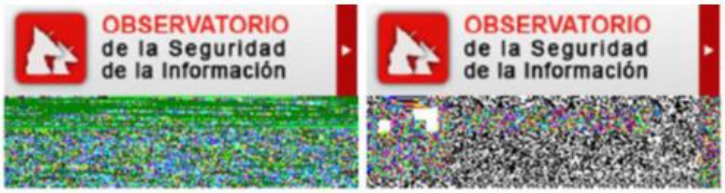
Como ya se ha mencionado, el estegoanálisis es la técnica que se usa para recuperar mensajes ocultos o para impedir la comunicación por esteganografía. Existen dos tipos principales de estegoanálisis pasivo, que se explican brevemente a continuación.

a) Estegoanálisis manual: Consiste en buscar de forma manual diferencias entre el *objeto contenedor* y el *estego-objeto* buscando cambios en la estructura para localizar datos ocultos. Los principales inconvenientes de esta técnica son que es necesario tener el objeto contenedor y que en muchas ocasiones se detecta que un objeto contiene información oculta pero es imposible recuperarla.

No obstante, cuando no se dispone del *archivo contenedor*, se pueden buscar irregularidades en el *archivo esteganografiado* para tratar de encontrar signos de la existencia de datos ocultos.

Los ataques visuales alertan al ojo humano de la presencia de información oculta gracias a la aplicación de filtros. Considérese el caso del BMP donde el bit menos significativo de las componentes de algunos de sus píxeles ha sido sustituido por información oculta. En tal escenario el estegoanálisis manual consiste en aplicar un filtro tal que sólo se considere el bit menos significativo de cada componente RGB de cada píxel. original.

Esto es lo que se ha hecho en la ilustración, la primera imagen oculta información y al aplicar el filtro salta a la vista un pequeño patrón uniforme en la parte superior de la imagen, además del cambio de tonalidad global con respecto a la imagen filtrada del archivo original.



Estas diferencias se deben a que la ocultación de información en LSB parte de la premisa de que la información almacenada originalmente en dicho bit es aleatoria. Esto no es cierto y la ocultación de información en él proporciona pistas adicionales a un analista. Precisamente por esto, las imágenes con poca variabilidad de colores y/o regiones uniformes son malas candidatas para una técnica de esteganografía LSB. Una imagen robusta ante un ataque de este tipo es una imagen natural no artificial con mucha variación de tonos y/o colores.

b) Estegoanálisis estadístico

Consiste en el cotejo de la frecuencia de distribución de colores del *estego-objeto*. Es una técnica lenta para la que se debe emplear software especializado. Estos programas suelen buscar pautas para ocultar los mensajes que utilizan los programas más habituales de esteganografía, este enfoque los hace muy eficaces cuando se trata de mensajes ocultos con estos programas típicos. Ahora bien, los mensajes ocultados manualmente son casi imposibles de encontrar para estos programas.

Una de estas técnicas es el ataque Chi-Square1 que permite estimar el tamaño de la posible información oculta en un *estego-objeto*. Es aplicable cuando un conjunto fijo de parejas de valores (PoVs) conmutan de un valor al otro de la pareja cuando se inserta los bits del mensaje oculto.

Otras Aplicaciones

*** Esteganografía empleando el protocolo TCP/IP**: El protocolo TCP/IP es apropiado para crear canales encubiertos de comunicación ya que a través de las cabeceras se pueden enviar datos relevantes para dos entes que acuerdan un protocolo encubierto. Usando este enfoque es posible empotrar datos en peticiones de conexión iniciales, conexiones establecidas u otros pasos intermedios.
Por ejemplo, considerando únicamente la cabecera TCP, se pueden ocultar datos en el número de secuencia inicial de una conexión. Esto ofrece 32 bits de datos ocultos por paquete de conexión inicial (SYN), es decir, 4 caracteres ASCII. Siguiendo esta filosofía se puede ocultar información en otros campos de las cabeceras de los distintos protocolos que componen TCP/IP, siempre y cuando los cambios no impliquen el rechazo de los paquetes intercambiados.
*** Control de malware:** El malware de hoy día normalmente se comunica con un punto de control en posesión del atacante para recibir órdenes de descarga de módulos adicionales, para enviar datos robados, para indicar que una nueva víctima ha sido infectada, etc.
El protocolo más utilizado para este tipo de comunicación es HTTP ya que normalmente tiene lugar sobre un puerto que no está filtrado por los cortafuegos y porque puede pasar desapercibido en el resto de tráfico de red generado por la navegación legítima.
La facilidad de establecer un canal de control con peticiones HTTP GET/POST tradicionales también implica que la comunicación es fácilmente detectable (si no se emplean técnicas de cifrado) por las empresas que gestionan los servidores web/servidores de DNS asociados al enlace de control. Aun más fácil es la identificación e interpretación de dicha comunicación para un analista de malware. Esto significa que ante denuncia de actividad ilegal asociada a un determinado punto de control, las infraestructuras son cerradas más rápidamente por las empresas que las gestionan.

*** Marcas de agua digitales**: Una marca de agua digital es un código de identificación que se introduce directamente en el contenido de un archivo multimedia, normalmente para incluir información relativa a los derechos de autor o de propiedad del contenido digital en cuestión.
La presencia de esta marca de agua debe ser inapreciable para el sistema de percepción humano a la vez que fácilmente extraíble por una aplicación que conozca el algoritmo para recuperarla.
Como consecuencia, en la actualidad se están empleando técnicas que en algunos casos pueden ser consideradas esteganografía para conseguir dicho propósito.
Las aplicaciones más comunes de las marcas de agua son:
• ***Prueba de propiedad***: identificación de la fuente, el autor, el propietario, el distribuidor y/o el consumidor de un archivo digital.
• ***Fingerprinting****:* incluye los datos asociados a una transacción, datos del propietario de un archivo y de su comprador. Permite identificar al responsable de copias ilegales de contenido protegido por derechos de autor.
• ***Clasificación de contenidos****: las* marcas de agua se pueden emplear apara indicar el tipo de contenido de un archivo. Por ejemplo, en un mundo ideal las páginas con contenido para adultos incluirían marcas de agua específicas en todas sus imágenes, vídeos y demás. Los programas de filtrado de contenidos las detectarían automáticamente de forma sencilla e impedirían su visualización a determinado público.
• ***Restricción en el uso de contenidos****:* empleadas en conjunción con aplicaciones o sistemas embebidos programados para tal fin, las marcas de agua pueden emplearse para impedir la visualización de ciertos contenidos cuando son copiados más de un determinado número de veces, a partir de una determinada fecha, etc.

En cualquier caso, una buena marca de agua debe resistir a cambios producidos en el archivo original, debe ser resistente a manipulaciones de la propia marca de agua, no debe ser humanamente perceptible, y debe tener un impacto mínimo sobre las propiedades estadísticas de su objeto contenedor. Dependiendo del uso que se vaya a hacer de la marca de agua también se puede señalar como propiedad deseada la facilidad para modificarla, por ejemplo para llevar la cuenta de cuántas veces se ha reproducido un determinado contenido.

Conclusiones

La esteganografía es una técnica en constante evolución, con una larga historia y con capacidad para adaptarse a nuevas tecnologías. A medida que las herramientas de esteganografía se hacen más avanzadas, las técnicas y las herramientas empleadas en el estegoanálisis también se hacen más complejas.

Los *archivos contenedores* no tienen que ser forzosamente imágenes, cualquier medio es válido (audio, video, ejecutables, etc.). Lejos de ser un ingenio exclusivamente teórico, el malware ha demostrado hacer un uso activo de la esteganografía, y se espera que surjan nuevos enfoques de uso para dificultar la labor de los analistas.

Pero las aplicaciones de esta ciencia no sólo se restringen al ámbito de lo poco ético, pudiendo ayudar en campos como la medicina, protección de menores, etc.

En cualquier caso, la eficiencia de las nuevas técnicas de estegoanálisis hace necesario el uso de la esteganografía combinada con criptografía con el fin de alcanzar un nivel de seguridad razonable. La criptografía garantiza la confidencialidad de una conversación pero no esconde el hecho de que dicha conversación se está manteniendo. Por otra parte, la esteganografía en solitario puede ocultar el hecho de que una conversación se mantiene, pero una vez descubierta la interacción, es posible que un atacante conozca el contenido intercambiado. Aun cuando descubrir el contenido original fuera difícil, un atacante puede modificar el estego-objeto para impedir la comunicación (ataque activo). Conjugando ambas técnicas se alcanza una complementariedad que multiplica la seguridad de un intercambio de mensajes.